CSE 3241 Project Checkpoint 01

Sam Bossley | Michael Izzo | Josephine Ko | Henry Xiong

1. List names of all your team members. Provide a paragraph explaining how you have been working as a team under remote setup so far, how you plan to communicate with each other, share work, etc. Any issues related to time differences, technology constraints, etc?

Our names are Henry Xiong, Sam Bossley, Josephine Ko, and Michael Izzo. We have been communicating over Discord primarily, and we plan to have weekly meetings, around 1-2 PM a week concerning the project, depending on project progress. We will share code work in a Github repository, and we plan to primarily use Discord like a work Slack. We do not have any other issues at this time.

NOTE

We, Josephine; Henry; and Sam, originally had a student from a different section of this course assigned to our group, but due to the nature of the course sections, he was consequently removed. However, we were not notified when a new member replacing the previous student was added, so we went ahead and worked on CP01 last week (9/10) and finished the ERD at the beginning of this week (9/14). It was not until Tuesday (9/15) during lecture that we were made aware of the situation; Michael had sent us an email regarding when we should meet to start CP01. Thus, Michael was unfortunately unable to contribute to this checkpoint. We will be doing our best to catch him up with what we have done before the next checkpoint is due.

2. Based on the requirements given in the project overview, list the entities to be modeled in this database. For each entity, provide a list of associated attributes. Make sure that your design allows for proper handling of buyer /seller interactions such as orders, payments, feedback, and karma points.

Table 1: Entities and Properties

ACCOUNT	BUYER ACCOUNT	SELLER ACCOUNT
- Name - Email - Phone number - Birthday	- Payment type listing - Shipping addresses	- Virtual store listing - Payment destination setup

- Address		

VIRTUAL STORE	IP ITEM	TRANSACTION
- Name - Description - Banner - Seller bio - Seller photo - URLs	- Title - Description - Price - Screenshot/Images - Keywords - File type - Type of accepted payment	- Payment type - Buyer account - Seller account - Item name - Virtual store name

3. Based on the requirements given in the project overview, what are the various relationships between entities?

(For example, "CUSTOMER entities purchase IP Item entities").

The relationships between the entities are as follows:

ACCOUNT is a superclass of subclasses SELLER ACCOUNT and BUYER ACCOUNT.

SELLER ACCOUNT entities accept TRANSACTIONs.

A TRANSACTION can contain multiple IP_ITEMs.

There can be any number of IP_ITEMs present on the VIRTUAL_INVENTORY.

A BUYER ACCOUNT can browse VIRTUAL_INVENTORYs.

4. Propose at least two additional entities that it would be useful for this database to model beyond the scope of the project requirements. Provide a list of possible attributes for the additional entities and possible relationships they may have with each other and the rest of the entities in the database. Give a brief, one sentence rationale for why adding these entities would be interesting/useful to the stakeholders for this database project.

Two additional entities that we could add are the Support Account and the Wishlist. The Support Account would be a subclass of the Account superclass, and would be connected to the other two accounts, and the properties it would have include ticket number, opening and closing date of the ticket, and notes regarding the ticket. The Wishlist would be connected to the Buyer Account and IP items and would have the properties of dates for each item, and other whether that item has been satisfied or not, like a boolean for each item. These entities would be useful

because a Support Account would help out other accounts and generally improve the user experience and with Wishlists accounts can view other people's accounts and buy their friends stuff from the Wishlist as well as Seller Accounts viewing Buyer Accounts' Wishlists and make sell offers.

5. Give at least four examples of some informal queries/reports that it might be useful for this database might be used to generate. Include one example for each of the additional entities you proposed in question 4 above.

One query could be a list of all transactions of every user in the past day, for general inventory and tracking sales. Another query might be similar, counting the number of sales for a single item in a day. Another useful metric to track would be a history list of all accounts that have received support in the past week, to determine which accounts have the most problems. Finally, using the wishlist entity to compile a list of all wishlist items any account has added would be helpful for the company to see what items users are interested in.

6. Suppose we want to add a new IP Item to the database. How would we do that given the entities and relationships you've outlined above? Is it possible to add up to five images for the IP Item? Is it possible for the IP Item to be purchased by more than one Payment Type? Is it possible for the Buyer to purchase IP Items from multiple Sellers at one time? Can a Buyer leave feedback on multiple items in the Seller's store? Explain how your model supports these possibilities. If it does not, make changes that allow your design to support all these requirements.

See revision on page 6

To add a new IP Item, we would need to add all the properties for an item, such as item images, then connect it to a virtual inventory. When a transaction is made, it will be linked to the specific IP Item. Since payment type is connected to a transaction, a buyer can use differing payment types for each transaction. A transaction can hold multiple items of different types, which map to virtual stores. For each IP Item, a user can leave reviews to show their support (or criticism) or the product.

7. Determine at least three other informal update operations and describe what entities would need to have attributes altered and how they would need to be changed given your above descriptions. Include one example for each of the additional entities you proposed in question 4 above.

Three other informal update operations would include updating account profile information, editing an order information (after the order has been made), and editing virtual store/inventory information.

In the first one, it would go for any account, so it would be the ACCOUNT superclass. Any one of its information can be altered after authentication and confirmation, like name, username, profile picture, and password.

For editing an order information, certain attributes off the TRANSACTIONs entity can be changed. For example, payment type and quantity are attributes that can be changed mid-order, but nothing else can be changed unless the transaction has been cancelled or completed.

The last one is editing virtual inventory information, where the affected entity is VIRTUAL INVENTORY. Any attribute on there can be edited, all the way from description to URLs to seller bio. Similar to changing information on an ACCOUNT, it would also go through similar authentication and confirmation. One example for the added SUPPORT ACCOUNT above would be like editing the profile picture of a user's account, going in and changing it would require a confirmation email at the very least. The example for the WISHLIST is that if a user wants to add a new item to the list, all the user has to do is to go to the page of the item, and then click and select which wishlist they can add to (a user can have multiple wishlists), and add it in.

8. Provide an ER diagram for your database. Make sure you include all of the entities and relationships you determined in the questions above INCLUDING the entities for question 4 above, and remember that EVERY entity in your model needs to connect to another entity in the model via some kind of relationship. You can use draw.io for your diagram. If drawing on paper, make sure that your drawing is clear and neat. Ensure that you use a proper notation and include a legend

Remember Joel

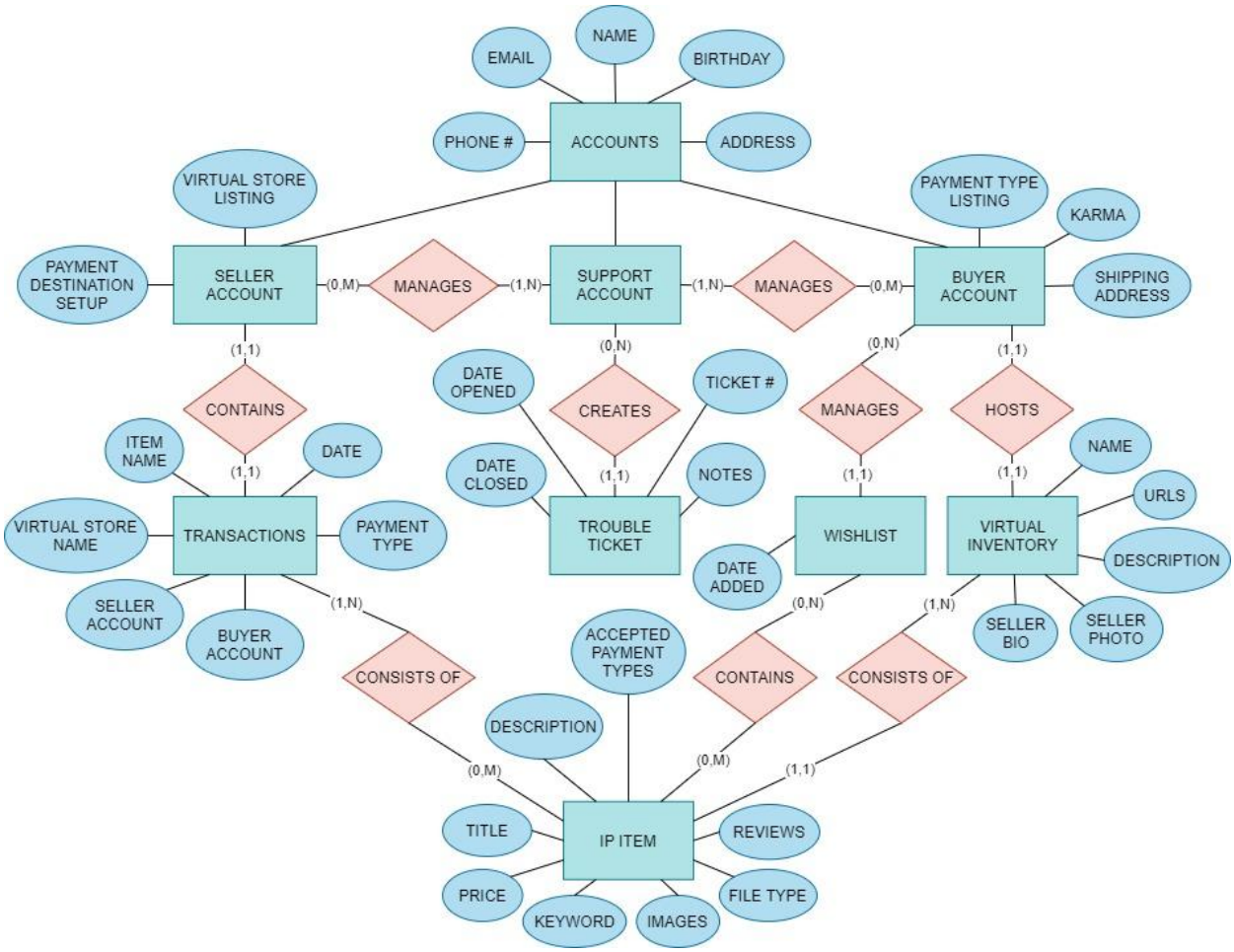
Table 2: ERD Legend

buyer account (1, 1)	.....	hosts a	.....	(1, 1) virtual inventory
buyer account (0, N)	....	manages	....	(1, 1) wishlist
seller account (1, N)	.....	contains	.....	(1, 1) transactions
support account (1, N)	....	manages	....	(0,M) seller account
support account (1, N)	....	manages	....	(0,M) buyer account
support account (0, N)	.....	creates	.....	(1,1) trouble ticket
transactions (1, N)	....	consists of	....	(0, M) ip items
virtual inventory (1, N)	.....	contains	.....	(1, 1) ip items
wishlist (0, N)	....	contains	....	(0, M) ip items

See Figure 1: Revised ERD v1.1 on page 7

(See diagram on following page)

Figure 1: ERD v1.0



CSE 3241 Project Checkpoint 01 Revisions

Sam Bossley | Michael Izzo | Josephine Ko | Henry Xiong

COMMENTS

6. IP ITEM has 0:5 IMAGE (or alternatively, images may be a special type of IP ITEM that is Listed) IP ITEM has 0:n PAYMENT TYPE SHOPPING CART has 1:n IP ITEM IP ITEM (not SELLER) has 0:n FEEDBACK

Heres a couple questions to ask yourself for your diagram: How will this DB handle Payment type? (buyers can purchase with karma points, CC, or Cryptocurrency) Multiple credit cards or payment types per customer? Multiple images per IP Item and multiple images per Seller account? Multiple items per purchase? From different Sellers? If B&B changes or adds an IP Type, how many entities in DB have to be changed?

ER Diagram ENTITIES (required) SELLER (emailID, store name, bannerImage, bio, sellerImage, Link1 BUYER (emailID, name, karma points) IP ITEM (name, description, price, payment type, image1, image2, image3, image4, image5) FEEDBACK (stars, comment) (recommended) SHOPPING CART (date, time) IMAGE (ID, Location) ITEM TYPE (description, file type) PAYMENT TYPE (description) CREDIT CARD (Number, Expiration Date, Buyer)

in this case since you dont have images as an entity, it should at least be a multivalued attribute since there are multiple instances of it

6. Suppose we want to add a new IP Item to the database. How would we do that given the entities and relationships you've outlined above? Is it possible to add up to five images for the IP Item? Is it possible for the IP Item to be purchased by more than one Payment Type? Is it possible for the Buyer to purchase IP Items from multiple Sellers at one time? Can a Buyer leave feedback on multiple items in the Seller's store? Explain how your model supports these possibilities. If it does not, make changes that allow your design to support all these requirements.

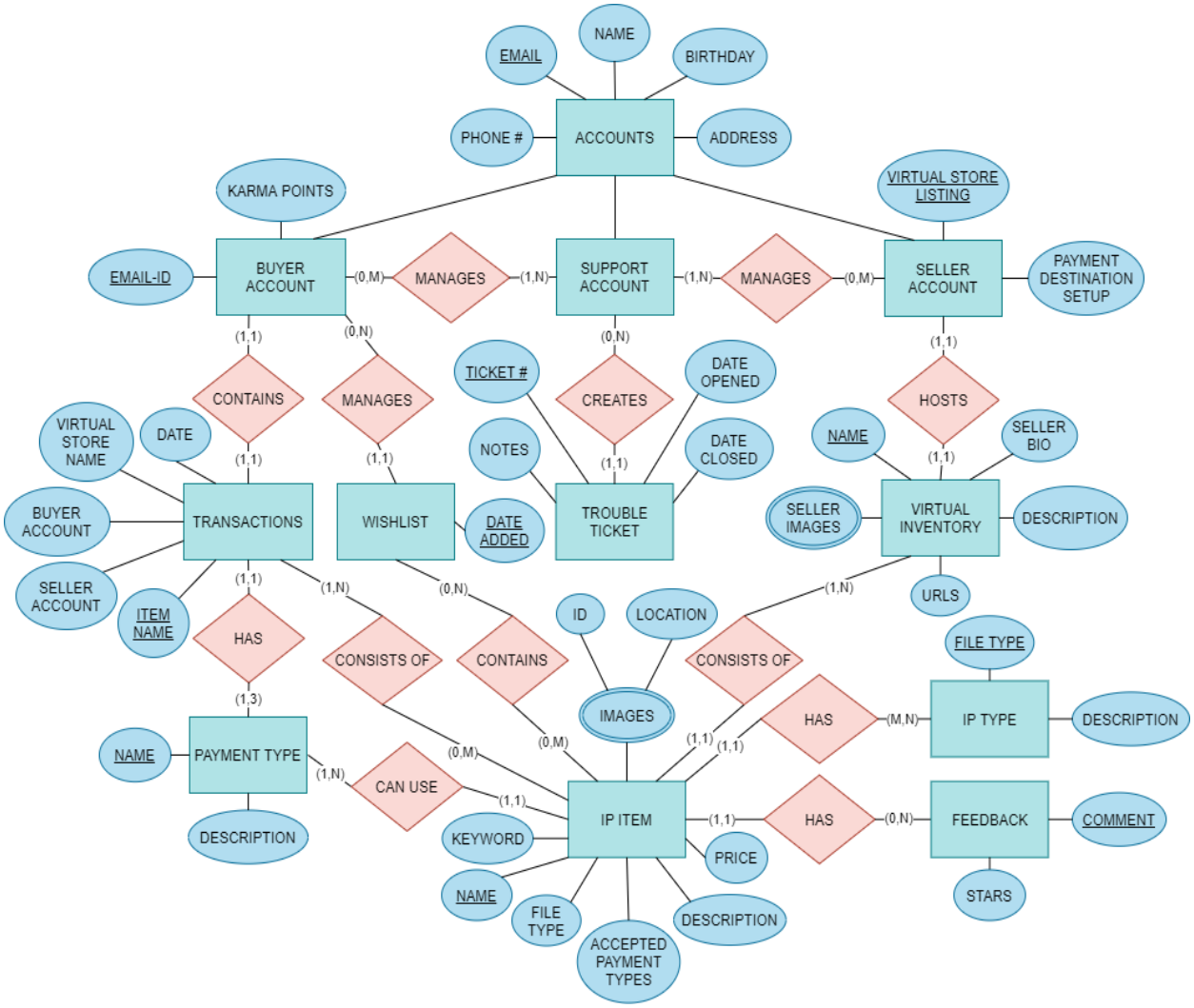
To add a new IP Item, we would need to add all the properties for an item, such as item images, then connect it to a virtual inventory. When a transaction is made, it will be linked to the specific IP Item. Since payment type is connected to a transaction, a buyer can use differing payment types for each transaction. A transaction can hold multiple items of different types, which map to virtual stores. For each IP Item, a user can leave reviews to show their support (or criticism) or the product.

It would be possible to add up from zero to five Images per IP Item, in that an IP Item does not need any Images, but only a maximum of five Images would be allowed for an IP Item. Any IP Item can be purchased by more than one Payment Type mainly because there can be any method of payment in this context from credit cards to Cryptocurrency. Alternatively, this would also allow for somebody to purchase something in partial Payment Types (e.g. paying half in

credit card, and paying the other half with PayPal.) It is possible for the Buyer to purchase multiple IP Items from multiple Sellers at once. A Shopping Cart for the Buyer does not necessarily have to contain only items from one Seller - a Buyer's Shopping Cart can contain any variety of items from any different Seller. Yes, a Buyer would be able to leave any amount of feedback on any number of IP Items in a Seller's store. For example, a Buyer doesn't have to leave any feedback on any IP Items in a Seller's store. Alternatively, a Buyer can also leave any number of feedback on any number of IP Items in a Seller's store.

8. Provide an ER diagram for your database.

Figure 1: Revised ERD v1.1

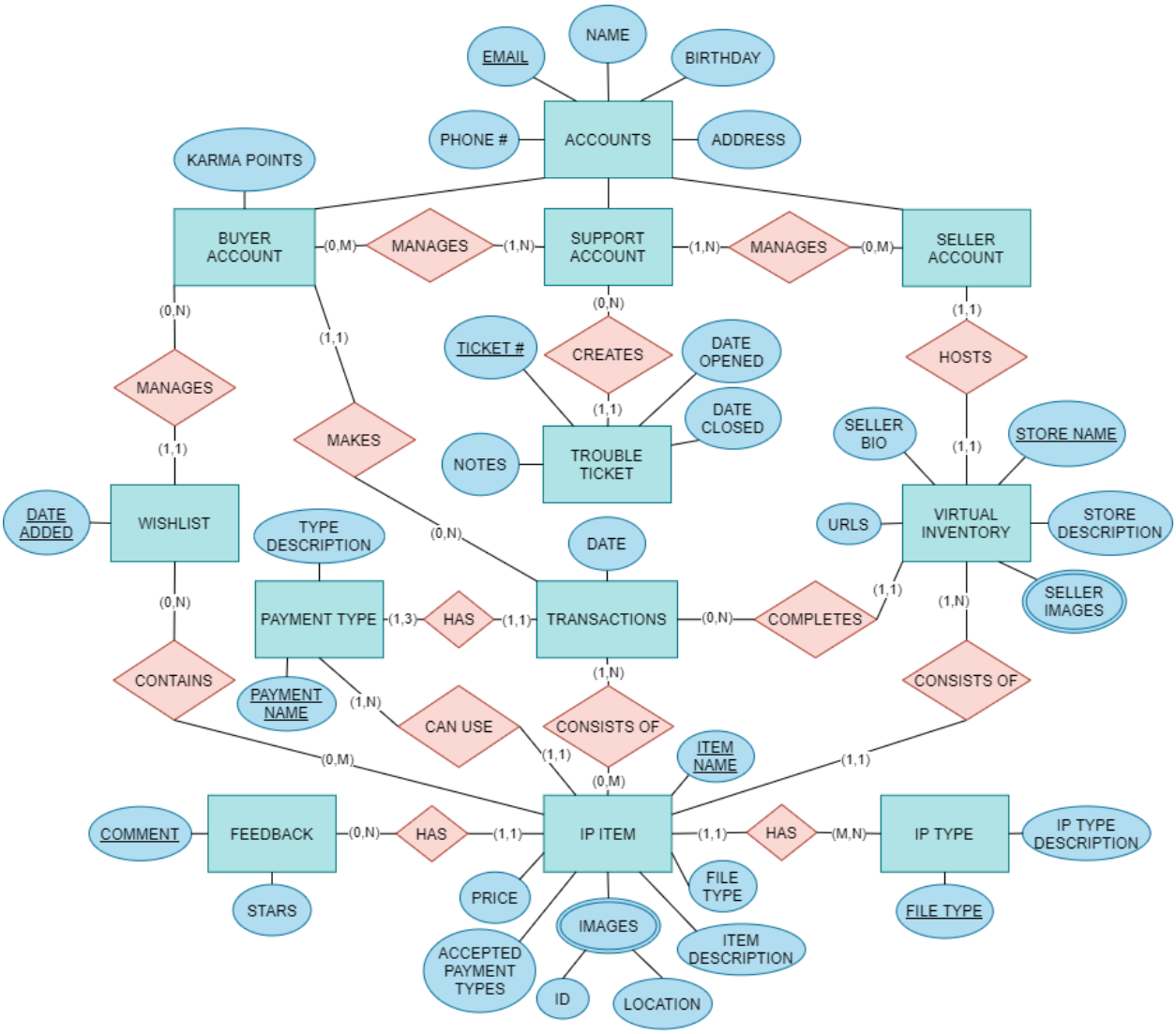


CSE 3241 Project Checkpoint 02

Sam Bossley | Michael Izzo | Josephine Ko | Henry Xiong

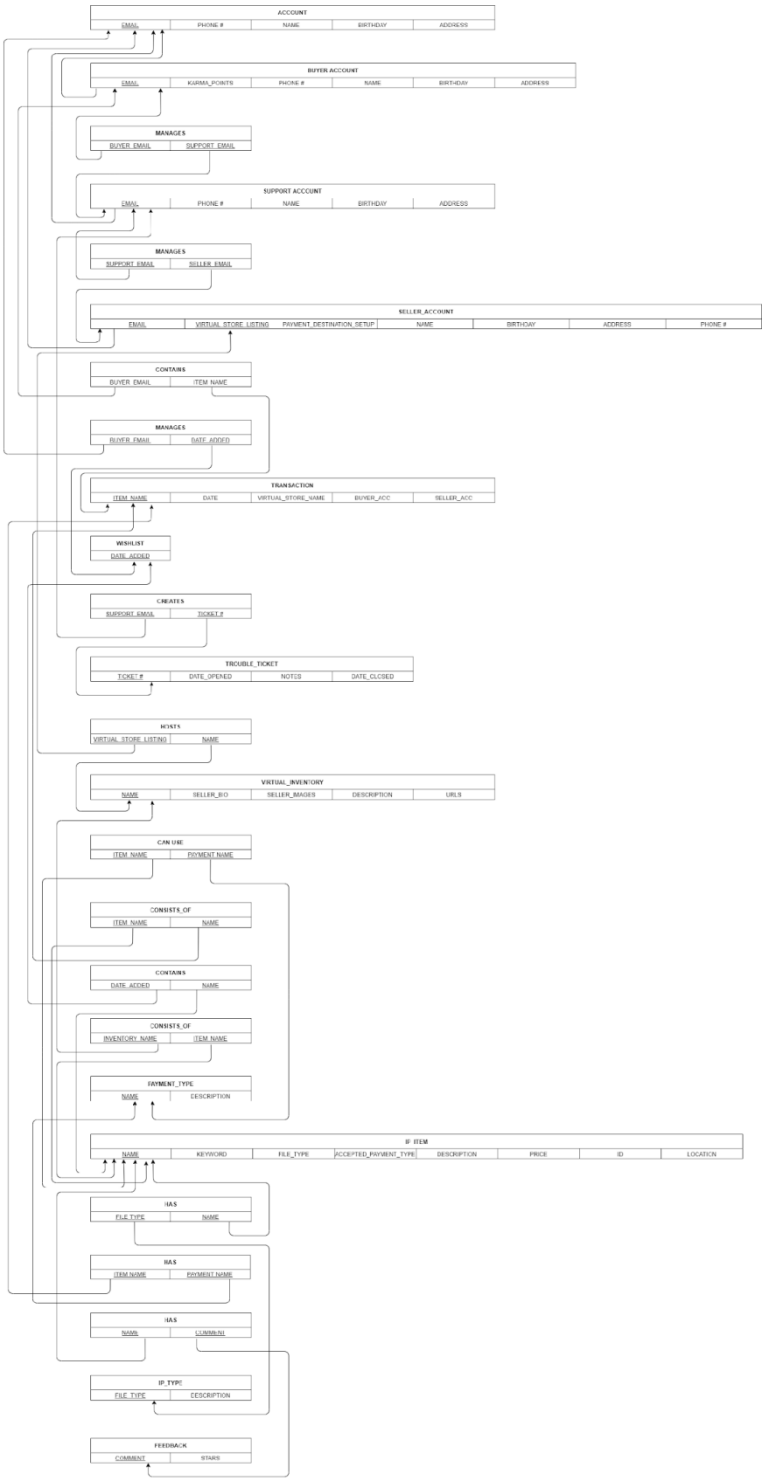
1. Provide a current version of your ER Model as per Project Checkpoint 01. If you were instructed to change the model for Project Checkpoint 01, make sure you use the revised version of your ER Model.

Figure 1: ERD v2.0



2. Map your ER model to a relational schema. Indicate all primary and foreign keys. ***Zoom in for full image***

Figure 2: Minimized Relational Schema v1.0



3. Given your relational schema, provide the relational algebra to perform the following queries. If your schema cannot provide answers to these queries, revise your ER Model and your relational schema to contain the appropriate information for these queries:

a. Find the titles of all IP Items by a given Seller that cost less than $10 (you choose how to designate the seller)

VIRT_INV ← (VIRTUAL_INVENTORY) ⋈ VIRTUAL_INVENTORY.NAME=SELLER_ACCOUNT.VIRTUAL_STORE_LISTING ($\sigma_{email=rememberjoel@gmail.com}$ SELLER_ACCOUNT)

VIRT_ITEMS ← (IP_ITEM) ⋈ IP_ITEM.VIRTUAL_INVENTORY=VIRT_INV.NAME AND IP_ITEM.PRICE < 10 (VIRT_INV)

π_{NAME} (VIRT_ITEMS)

b. Give all the titles and their dates of purchase made by given buyer (you choose how to designate the buyer)

BUYER ← ($\sigma_{email=rememberjoel@gmail.com}$ BUYER_ACCOUNT)

ALL_ITEMS ← (TRANSACTIONS) ⋈ TRANSACTIONS.BUYER_ACCOUNT=BUYER.EMAIL (BUYER)

$\pi_{ITEM\_NAME,DATE}$ (ALL_ITEMS)

c. Find the seller names for all sellers with less than 5 IP Items for sale

SELLER_INV ← (VIRTUAL_INVENTORY) ⋈ VIRTUAL_INVENTORY.NAME=SELLER_ACCOUNT.VIRTUAL_STORE_LISTING (SELLER_ACCOUNT)

SELLER_ITEMS ← (CONSISTS_OF) ⋈ CONSISTS_OF.VIRTUAL_INVENTORY_ID=SELLER_INV.NAME (SELLER_INV)

SELLER_ITEMS2 ← (SELLER_ITEMS) ⋈ SELLER_ITEMS..IP_ITEM_ID=IP_ITEM.NAME (IP_ITEM)

SELLER_ITEMS3(IP_ITEM_NAME, SELLER_NAME) ← $\pi_{IP\_ITEM.NAME,SELLER.NAME}$ (SELLER_ITEMS2)

ITEMS_SALE ← $\Im$ COUNT(SELLER_NAME) (SELLER_ITEMS3)

σ COUNT(SELLER_NAME) < 5 (ITEMS_SALE)

d. Give all the buyers who purchased an IP Item by a given seller (XXX) and the names of the IP Items they purchased

$\sigma_{SELLER\_ACCOUNT = XXX}$ TRANSACTIONS

$\pi_{BUYER\_ACCOUNT, ITEM\_NAME}$ TRANSACTIONS

π_{ITEM_NAME} (BUYERS)

e. Find the total number of IP Items purchased by a single buyer (XXX) (you choose how to designate the buyer)

$\sigma_{(BUYER\_ACCOUNT = XXX)}$ (TRANSACTIONS)

$\Im$COUNT(*) (π_{ITEM_NAME} (TRANSACTIONS))

f. Find the buyer who has purchased the most IP Items and the total number of IP Items they have purchased

$\sigma_{(BUYER\_ACCOUNT = XXX)}$ (TRANSACTIONS)

ITEMS_PURCH ← $\Im$COUNT(*) (π_{ITEM_NAME} (TRANSACTIONS))

$\Im_{MAX\ COUNT*}$ (ITEMS_PURCH)

4. Three additional interesting queries in plain English and also relational algebra. Your queries should include at least one of these:

a. outer joins

b. aggregate function

c. “extra” entities from CP01

a. Get a listing of all buyer accounts and seller accounts

USER_ACCOUNTS(buyer_name, seller_name) ← $_{BUYER\_ACCOUNT.name,\ SELLER\_ACCOUNT.name}$
((BUYER_ACCOUNTS) $\bowtie$ (SELLER_ACCOUNTS))

b. Find a virtual store’s most expensive listing

VIRT_STORE ← (VIRTUAL_INVENTORY) ⋈
VIRTUAL_INVENTORY.NAME=SELLER_ACCOUNT.VIRTUAL_STORE_LISTING (σ email=XXX SELLER_ACCOUNT)

ALL_ITEMS ← (σ VIRTUAL_INVENTORY.id=VIRT_STORE.id CONSISTS_OF)

π IP_NAME, PRICE (ALL_ITEMS)

ℑ MAX PRICE (ALL_ITEMS)

c. **Find all accounts that a support account has managed**

SUPPT_ACCT ← (σ email=XXX SUPPORT_ACCOUNT)

BUYERS_MANAGED ← (σ SUPPT_ACCT_ID=SUPPT_ACCT.email $MANAGES_{support\_to\_buyer}$)

SELLERS_MANAGED ← (σ SUPPT_ACCT_ID=SUPPT_ACCT.email $MANAGES_{support\_to\_seller}$)

ACCOUNTS_MANAGED ← BUYERS_MANAGED ⋈ SELLERS_MANAGED

CSE 3241 Project Checkpoint 03

Sam Bossley | Michael Izzo | Josephine Ko | Henry Xiong

1. Provide a current version of your ER Diagram and Relational Model as per Project Checkpoint 02. If you were instructed to change the model for Project Checkpoint 02, make sure you use the revised versions of your models

COMMENTS

Don't see any personally, but be sure to have have any FKs or surrogate keys in ERD. Be sure your db can handle: - Buyers purchasing with karma points, CC, or Cryptocurrency) -Multiple credit cards/payment types per buyer -Multiple images per IP Item/multiple images per Seller account? -Multiple items per purchase and from different Sellers -If B&B changes or adds an IP Type, how many entities in DB have to be changed? Person, Seller, Buyer should be a hierarchy, overlapping and total.

4. I like your queries here, these can provide some good info for your db

be sure to not have*

SIMPLIFIED

Be sure your DB can handle:

- Buyers purchasing with karma points, CC, or Cryptocurrency)
- Multiple credit cards/payment types per buyer
- Multiple images per IP Item/multiple images per Seller account?
- Multiple items per purchase and from different Sellers
- If B&B changes or adds an IP Type, how many entities in DB have to be changed?
- Person, Seller, Buyer should be a hierarchy, overlapping and total.

We simplified and corrected the ERD significantly. Only some of the revisions were based on grader comments.

(See diagrams and revisions on following pages)

Figure 1: ERD v2.0

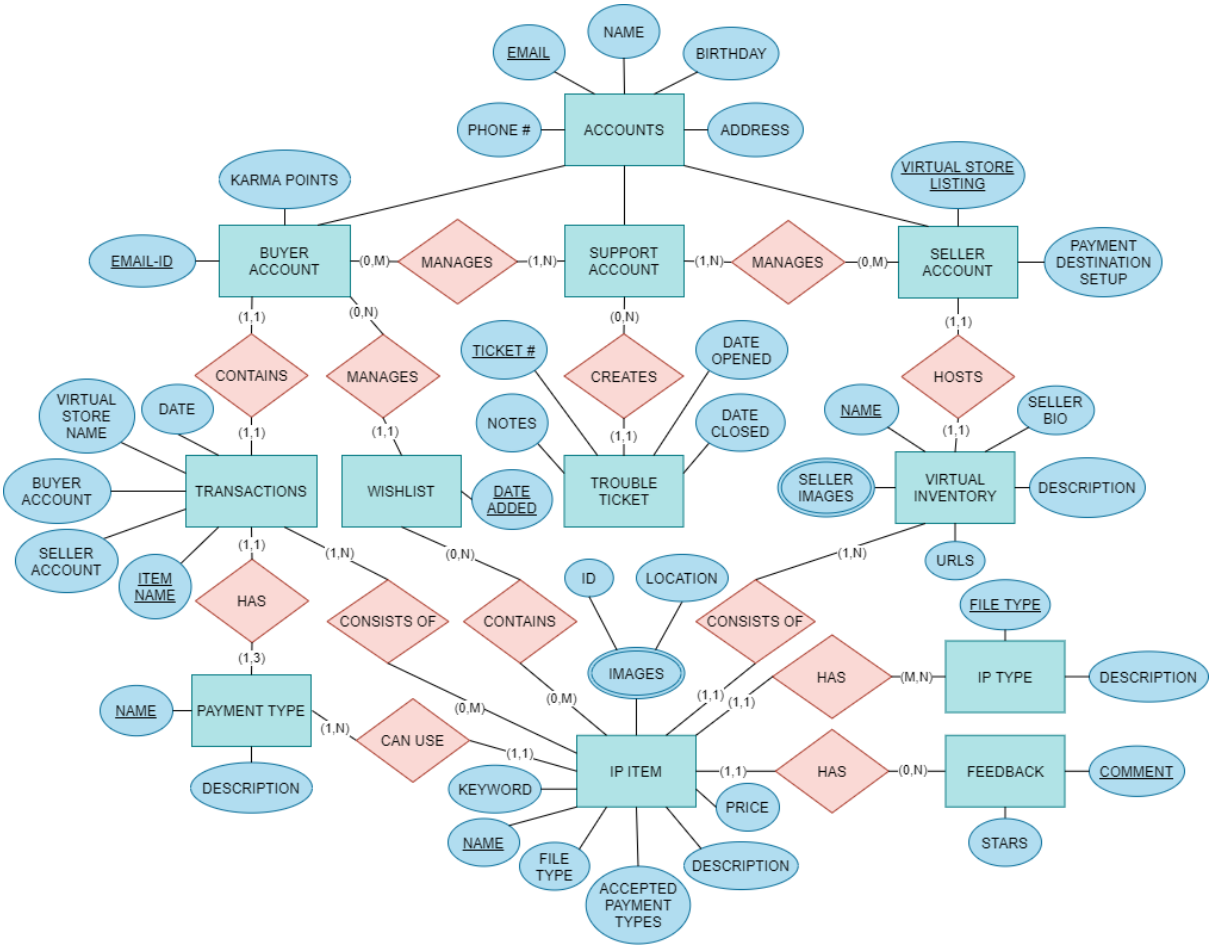
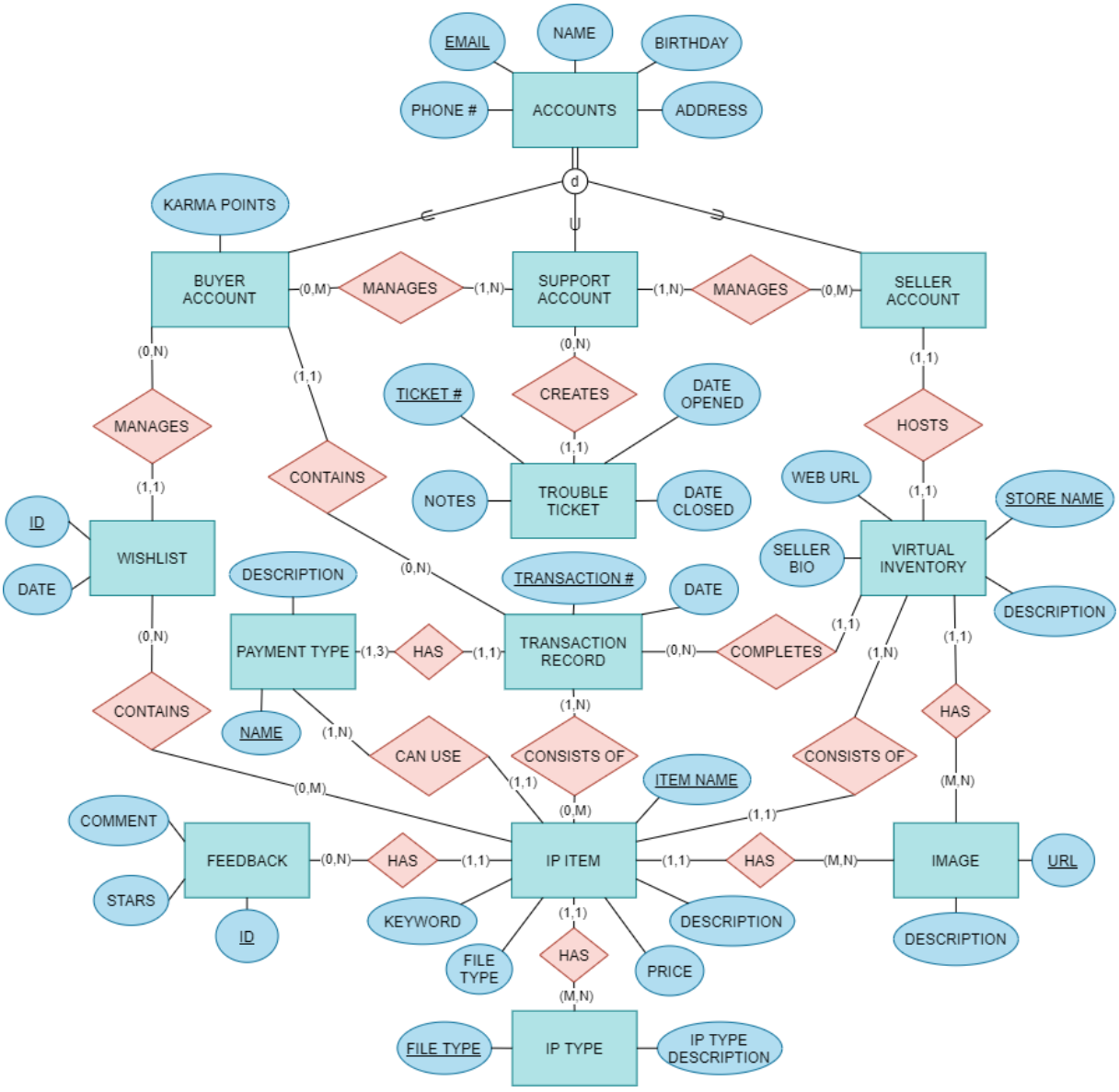


Figure 2: Revised ERD v3.0



2. Given your relational schema, create a text file containing the SQL code to create your database schema. Use this SQL to create a database in SQLite. Populate this database with the data provided for the project as well as 20 sample records for each table that does not contain data provided in the original project documents.

```
CREATE TABLE ACCOUNT (
   Email VARCHAR(30) PRIMARY KEY,
   Phone_num CHAR(9) NOT NULL,
   Name VARCHAR(25) NOT NULL,
   Birthday DATE NOT NULL,
   Address VARCHAR(40) NOT NULL
);

CREATE TABLE BUYER_ACCOUNT (
   Email VARCHAR(30) PRIMARY KEY,
   Phone_num CHAR(9) NOT NULL,
   Name VARCHAR(25) NOT NULL,
   Birthday DATE NOT NULL,
   Address VARCHAR(40) NOT NULL,
   Karma_points INT NOT_NULL
);

CREATE TABLE SUPPORT_MANAGES_BUYER (
   Buyer_email VARCHAR(30) NOT NULL,
   Support_email VARCHAR(30) NOT NULL,
FOREIGN KEY (Support_email) REFERENCES SUPPORT_ACCOUNT(Email)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Buyer_email) REFERENCES BUYER_ACCOUNT(Email)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE SUPPORT_ACCOUNT (
   Email VARCHAR(30) PRIMARY KEY,
   Phone_num CHAR(9) NOT NULL,
   Name VARCHAR(25) NOT NULL,
   Birthday DATE NOT NULL,
   Address VARCHAR(40) NOT NULL
);

CREATE TABLE SUPPORT_MANAGES_SELLER (
   Support_email VARCHAR(30) NOT NULL,
   Seller_email VARCHAR(30) NOT NULL,
FOREIGN KEY (Support_email) REFERENCES SUPPORT_ACCOUNT(Email)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Seller_email) REFERENCES SELLER_ACCOUNT(Email)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE SELLER_ACCOUNT (
   Email VARCHAR(30) PRIMARY KEY,
   Store_name VARCHAR(30) NOT NULL,
   Phone_num CHAR(9) NOT NULL,
   Name VARCHAR(25) NOT NULL,
   Birthday DATE NOT NULL,
   Address VARCHAR(40) NOT NULL,
FOREIGN KEY (Store_name) REFERENCES VIRTUAL_INVENTORY(Store_name)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE BUYER_CONTAINS_TRANSACTION_RECORDS (
   Buyer_email VARCHAR(30) NOT NULL,
```

```
   Transaction_num VARCHAR(30) NOT NULL,
FOREIGN KEY (Buyer_email) REFERENCES BUYER_ACCOUNT(Email)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Transaction_num) REFERENCES
TRANSACTION_RECORD(Transaction_num)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE BUYER_MANAGES_WISHLIST (
   Buyer_email VARCHAR(30) NOT NULL,
   Id INT NOT NULL,
FOREIGN KEY (Buyer_email) REFERENCES BUYER_ACCOUNT(Email)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Id) REFERENCES WISHLIST(Id)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE TRANSACTION_RECORD (
   Transaction_num VARCHAR(30) PRIMARY KEY,
   Date DATE NOT NULL,
   Store_name VARCHAR(30) NOT NULL,
   Item_name  VARCHAR(40) NOT NULL,
FOREIGN KEY (Store_name) REFERENCES VIRTUAL_INVENTORY(Store_name)
        ON DELETE CASCADE ON UPDATE CASCADE
FOREIGN KEY (Item_name) REFERENCES IP_ITEM(Item_name)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE WISHLIST (
   Id INT PRIMARY KEY,
```

```
    Date_added DATE NOT NULL
);

CREATE TABLE SPT_ACCT_CREATES_TICKET (
    Support_email VARCHAR(30) NOT NULL,
    Ticket_num INT NOT NULL,
FOREIGN KEY (Support_email) REFERENCES SUPPORT_ACCOUNT(Email)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Ticket_num) REFERENCES TROUBLE_TICKET(Ticket_num)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE TROUBLE_TICKET (
    Ticket_num INT PRIMARY KEY,
    Date_opened DATE NOT NULL,
    Date_closed DATE NOT NULL,
    Notes VARCHAR(50)
);

CREATE TABLE SELL_ACCT_HOSTS_INV (
    Store_name VARCHAR(30) NOT NULL,
    Seller_email VARCHAR(30) NOT NULL,
FOREIGN KEY (Store_name) REFERENCES VIRTUAL_INVENTORY(Store_name)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Seller_email) REFERENCES SELLER_ACCOUNT(Email)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE VIRTUAL_INVENTORY (
    Store_name VARCHAR(30) PRIMARY KEY,
```

```
    Seller_bio VARCHAR(280),
    Description VARCHAR(280),
    Web_url VARCHAR(100) NOT NULL
);

CREATE TABLE ITEM_CANUSE_PAYMENT_TYPE (
    Item_name VARCHAR(40) NOT NULL,
    Payment_name VARCHAR(10) NOT NULL,
FOREIGN KEY (Item_name) REFERENCES IP_ITEM(Item_name)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Payment_name) REFERENCES PAYMENT_TYPE(Name)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS (
    Item_name VARCHAR(40) NOT NULL,
    Transaction_num INT NOT NULL,
FOREIGN KEY (Item_name) REFERENCES IP_ITEM(Item_name)
        ON DELETE CASCADE ON UPDATE CASCADE
FOREIGN KEY (Transaction_num) REFERENCES
TRANSACTION_RECORD(Transaction_num)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE WISHLIST_CONTAINS_IP_ITEM (
    Id INT NOT NULL,
    Item_name VARCHAR(40) NOT NULL,
FOREIGN KEY (Id) REFERENCES WISHLIST(Id)
        ON DELETE CASCADE ON UPDATE CASCADE
FOREIGN KEY (Item_name) REFERENCES IP_ITEM (Item_name)
```

```
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE INV_CONSISTSOF_IP_ITEMS (
   Store_name VARCHAR(30) NOT NULL,
   Item_name VARCHAR(40) NOT NULL,
FOREIGN KEY (Store_name) REFERENCES VIRTUAL_INVENTORY(Store_name)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Item_name) REFERENCES IP_ITEM(Item_name)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE PAYMENT_TYPE (
   Name VARCHAR(10) PRIMARY KEY,
   Description VARCHAR(20) NOT NULL
);

CREATE TABLE IP_ITEM (
   Item_name VARCHAR(40) PRIMARY KEY,
   Keyword VARCHAR(20) NOT NULL,
   File_type VARCHAR(20) NOT NULL,
   Description VARCHAR(100) NOT NULL,
   Price FLOAT NOT NULL
);

CREATE TABLE IP_ITEM_HAS_IP_TYPE (
   File_type VARCHAR(20) NOT NULL,
   Item_name INT NOT NULL,
FOREIGN KEY (File_type) REFERENCES IP_TYPE(File_type)
        ON DELETE CASCADE ON UPDATE CASCADE
```

```
FOREIGN KEY (Item_name) REFERENCES IP_ITEM(Item_name)
        ON DELETE CASCADE ON UPDATE CASCADE,
);

CREATE TABLE TRANSACTION_RECORD_HAS_PAYMENT_TYPE (
   Transaction_num INT NOT NULL,
   Payment_name VARCHAR(20) NOT NULL,
FOREIGN KEY (Transaction_num) REFERENCES
TRANSACTION_RECORD(Transaction_num)
        ON DELETE CASCADE ON UPDATE CASCADE
FOREIGN KEY (Payment_name) REFERENCES PAYMENT_TYPE(Payment_name)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE IP_TYPE (
   File_type VARCHAR(20) PRIMARY KEY
   IP_type_description VARCHAR(200) NOT NULL,
);

CREATE TABLE FEEDBACK (
   Id INT PRIMARY KEY,
   Stars INT NOT NULL,
   Item_name VARCHAR(40),
   Comment VARCHAR(500) NOT NULL,
FOREIGN KEY (Item_name) REFERENCES IP_ITEM(Item_name)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE INV_HAS_IMAGES (
   Store_name VARCHAR(30) NOT NULL,
```

```
   Url VARCHAR(50) NOT NULL,
FOREIGN KEY (Store_name) REFERENCES VIRTUAL_INVENTORY(Store_name)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Url) REFERENCES IMAGE(Url)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE IP_ITEM_HAS_IMAGES (
   Item_name VARCHAR(40) NOT NULL,
   Url VARCHAR(50) NOT NULL,
FOREIGN KEY (Item_name) REFERENCES IP_ITEM(Item_name)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (Url) REFERENCES IMAGE(Url)
        ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE IMAGE (
   Url VARCHAR(50) PRIMARY KEY,
   Description VARCHAR(50) NOT NULL
);
```

#

populating db

#

```
INSERT INTO BUYER_ACCOUNT
VALUES (
   "thelegend27@gmail.com",
   "5024119782",
   "John Ross",
   "1981-07-01",
   "1234 W Fake St, Columbus OH 43210",
   0
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "jontron@gmail.com",
   "6146952553",
   "Jon Tresko",
   "2001-04-12",
   "4241 Hehe Pt, Detroit, MI 49201",
   26
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "billship3@gmail.com",
   "2487833663",
   "Bill Ship",
   "1999-05-13",
   "8568 Circle Pt, Northville, MI 48168",
   90
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "LilySmith4@gmail.com",
   "8346333467",
   "Lily Smith",
   "1989-12-13",
   "58932 Square Dr, Buffalo, NY 14042",
   45
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "PlaneJane5@gmail.com",
   "3014012467",
   "Jane Goodal",
   "1945-01-11",
   "554 Ocean Dr, Miami, FL 33139",
   0
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "GeorgeLucas6@gmail.com",
   "7013202194",
   "George Lucas",
   "1970-30-18",
   "90210 Rodeo Dr, Beverly Hills, CA 90210",
   16
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "Bubbawatson7@gmail.com",
   "6654032102",
   "Bubba Watson",
   "1983-02-13",
   "767 Golf Course Dr, Augusta, GA 30805",
   9
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "BarackObama8@gmail.com",
   "8881010123",
   "Barack Obama",
   "1961-08-04",
   "1600 Pennsylvania Ave NW, Washington, DC 20500",
   44
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "MichelleObama9@gmail.com",
    "8881010124",
   "Michelle Obama",
   "1964-01-17",
   "1600 Pennsylvania Ave NW, Washington, DC 20500",
   44
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "MattJones10@gmail.com",
   "550132748",
   "Matt Jones",
   "1999-10-05",
   "9320 Rectangle Dr, Ft Myers, FL 33132",
   0
);


INSERT INTO BUYER_ACCOUNT
VALUES (
   "AmongUs11@gmail.com",
   "8874320011",
   "Among Us",
```

```
    "2020-09-01",
    "88430, 2010 N High St, Columbus, OH 43201",
    4
);


INSERT INTO BUYER_ACCOUNT VALUES (
    "DrewToo12@gmail.com",
    "8909305531",
    "Drew Michael",
    "1993-01-10",
    "99601 Lakeview Dr, Buffalo, NY 14042",
    9
);


INSERT INTO BUYER_ACCOUNT VALUES (
    "GeneSith13@gmail.com",
    "6142038520",
    "Gene Smith",
    "1989-12-13",
    "873 Dubin Rd, Dublin, OH 43016",
    45
);


INSERT INTO BUYER_ACCOUNT VALUES (
    "BooneJenner14@gmail.com",
    "6144033301",
    "Boone Jenner",
    "1990-12-03",
    "1023 N High St, Columbus, OH 43205",
    38
);


INSERT INTO BUYER_ACCOUNT VALUES (
    "BluesClues15@gmail.com",
    "8887300987",
    "Blues Clue",
    "1983-03-03",
    "1700 Clues Dr, Boise, ID 31042",
    19
);


INSERT INTO BUYER_ACCOUNT VALUES (
    "GeorgeWashington16@gmail.com",
    "9990134234",
    "George Washington",
    "1930-12-01",
    "78422 Arlington Dr, Alrington, VA 14042",
    1
);


INSERT INTO BUYER_ACCOUNT VALUES (
    "CharlesButt17@gmail.com",
    "4455930875",
    "Charles Butt",
    "1980-06-14",
    "4320 Tree Rd, San Francisco, CA 90123",
    4
);


INSERT INTO BUYER_ACCOUNT VALUES (
    "CharlesJones18@gmail.com",
    "3456049032",
    "Charles Jones",
    "1992-05-13",
    "789 Pond Dr, Dallas, TX 33042",
    100
);


INSERT INTO BUYER_ACCOUNT VALUES (
    "ChrisMerit19@gmail.com",
    "7345409707",
    "Chris Merit",
    "1970-06-01",
    "44912 Cypress Dr, Detroit, MI 48169",
    16
);


INSERT INTO BUYER_ACCOUNT VALUES (
    "CharlotteWeb20@gmail.com",
    "7046503203",
    "Charlotte Web",
    "1960-04-13",
    "730 Farm Dr, Farmville, NY 90432",
    15
);


INSERT INTO SUPPORT_MANAGES_BUYER VALUES (
    "mike.123@bnb.com",
    "thelegend27@gmail.com"
);


INSERT INTO SUPPORT_MANAGES_BUYER VALUES (
    "Matt.2@bnb.com",
    "CharlotteWeb20@gmail.com"
);


INSERT INTO SUPPORT_MANAGES_BUYER VALUES (
    "Guess.3@bnb.com",
    "CharlesJones18@gmail.com"
);
```

```
INSERT INTO
SUPPORT_MANAGES_BUYER VALUES
(
   "Kylie.4@bnb.com",
   "CharlesButt17@gmail.com"
);

INSERT INTO
SUPPORT_MANAGES_BUYER VALUES
(
   "Taylor.5@bnb.com",
   "ChrisMerit19@gmail.com"
);

INSERT INTO
SUPPORT_MANAGES_BUYER VALUES
(
   "Dan.6@bnb.com",
   "GeorgeWashington16@gmail.com"
);

INSERT INTO
SUPPORT_MANAGES_BUYER VALUES
(
   "Olivia.7@bnb.com",
   "BluesClues15@gmail.com"
);

INSERT INTO
SUPPORT_MANAGES_BUYER VALUES
(
   "Jimmy.8@bnb.com",
   "BooneJenner14@gmail.com"
);

INSERT INTO
SUPPORT_MANAGES_BUYER VALUES
(
   "Liam.9@bnb.com",
   "BarackObama8@gmail.com"
);

INSERT INTO
SUPPORT_MANAGES_BUYER VALUES
(
   "Emma.10@bnb.com",
   "billship3@gmail.com"
);

INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "mike.123@bnb.com",
   "5024119782",
   "Mike Young",
   "1972-10-22",
   "3379 3rd St, Franklin, OH 45902"
);

INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Matt.2@bnb.com",
   "5024119783",
   "Matt Smith",
   "1971-09-21",
   "3379 3rd St, Franklin, OH 45902"
);

INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Guess.3@bnb.com",
   "5024119784",
   “Guess lewis",
   "1962-02-11",
   "3379 3rd St, Franklin, OH 45902"
);

INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Kylie.4@bnb.com",
   "5024119785",
   "Kylie Jenner",
   "1981-04-11",
   "3379 3rd St, Franklin, OH 45902"
);

INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Taylor.5@bnb.com",
   "5024119786",
   "Taylor Swift",
   "1990-04-01",
   "3379 3rd St, Franklin, OH 45902"
);

INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Dan.6@bnb.com",
   "5024119787",
   "Dan Jones",
   "1974-09-61",
   "3379 3rd St, Franklin, OH 45902"
);

INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Olivia.7@bnb.com",
   "5024119788",
   "Olivia Newton",
   "1990-08-31",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Jimmy.8@bnb.com",
   "5024119789",
   "Jimmy Franklin",
   "1995-01-07",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Liam.9@bnb.com",
   "5024119790",
   "Liam Blackwell",
   "1989-10-09",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Emma.10@bnb.com",
   "5024119791",
   "Emma Watson",
   "1988-12-25",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Ava.11@bnb.com",
   "5024119792",
   "Ava Martin",
   "1993-06-30",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Sophia.12@bnb.com",
   "5024119793",
   “Sophia Norris",
   "1978-09-05",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "William.13@bnb.com",
   "5024119794",
   "William Washington",
   "1990-07-20",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Oliver.14@bnb.com",
   "5024119795",
   "Oliver Westing",
   "1995-02-01",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Asher.15@bnb.com",
   "5024119796",
   "Asher Wood",
   "1999-01-12",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Brutus.16@bnb.com",
   "5024119797",
   "Brutus Buckeye",
   "1950-01-01",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Justin.17@bnb.com",
   "5024119798",
   "Justin Fields",
   "1986-05-12",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "Luke.18@bnb.com",
   "5024119799",
   "Luke Skywalker",
   "1974-09-15",
   "3379 3rd St, Franklin, OH 45902"
);
```

```
INSERT INTO SUPPORT_ACCOUNT
VALUES (
```

```
   "Julius.19@bnb.com",
   "5024119100",
   “Julius Caesar",
   "1990-07-01",
   "3379 3rd St, Franklin, OH 45902"
);


INSERT INTO SUPPORT_ACCOUNT
VALUES (
   "George.20@bnb.com",
   "5024119101",
   "George Kelly",
   "1973-09-23",
   "3379 3rd St, Franklin, OH 45902"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "mike.123@bnb.com",
    "google@gmail.com"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "Ava.11@bnb.com",
    "google@gmail.com"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "Sophia.12@bnb.com",
    "Florida@gmail.com"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "William.13@bnb.com",
    "New York@gmail.com"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "Oliver.14@bnb.com",
    "Texas@gmail.com"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "Asher.15@bnb.com",
    "Texas@gmail.com"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "Brutus.16@bnb.com",
    "Computer@gmail.com"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "Justin.17@bnb.com",
    "Ohio@gmail.com"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "Luke.18@bnb.com",
    "Clocks@gmail.com"
);


INSERT INTO
SUPPORT_MANAGES_SELLER VALUES
(
   "Julius.19@bnb.com",
    "TV@gmail.com"
);


INSERT INTO SELLER_ACCOUNT
VALUES (
   "google@gmail.com",
   "4997302322",
   "Google",
   "1999-04-30",
   "Google HQ, Cool City, CA 54028",
   "Google Store"
);


INSERT INTO SELLER_ACCOUNT
VALUES (
   "youtuber@gmail.com",
   "4997302325",
   "Youtube",
   "2005-05-20",
   "Youtube HQ, Cooler City, CA 54029",
   "Youtube Store"
);


INSERT INTO SELLER_ACCOUNT
VALUES (
   "Florida@gmail.com",
   "8659348890",
   "Florida",
   "1950-01-30",
   "32 Atlantic Rd, Dunedin, FL 34698",
   “Florida Store"
);


INSERT INTO SELLER_ACCOUNT
VALUES (
   "New York@gmail.com",
   "9057869043",
   "New York”,
   "1959-03-09",
   "170 Pilgrim Street Mahopac, NY
10541",
   "New York Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "NorthCarolina@gmail.com",
  “7045679034",
   "North Carolina",
   "1998-05-28",
   "845 Cedar Ave, Fuquay Varina, NC 27526",
   "North Carolina Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "Texas@gmail.com",
   "889532143",
   "Texas",
   "2004-04-09",
   "7087 Carson Drive, Euless TX 76039",
   "Texas Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "google2@gmail.com",
   "4997302323",
   "Google2",
   "1999-04-31",
   "Google HQ2, Cool City2, CA 54028",
   "Google Store 2"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "google3@gmail.com",
   "4997302325",
   "Google3",
   "1999-05-01",
   "Google HQ3, Cool City3, CA 54028",
   "Google Store 3"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "Gmail@gmail.com",
   "4997302567",
   "Gmail",
   "2003-12-30",
   "Google HQ, Cool City, CA 54028",
   "Gmail Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "Ohio@gmail.com",
   "6143988892",
   "Ohio",
   "1934-12-26",
   "221 Hickory St, Tiffin, OH 44883",
   "Ohio Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "Alabama@gmail.com",
   "4998390134",
   "Alabama",
   "2001-04-07",
   "7052 Bellevue Drive, Pelham, AL 35124",
   "Alabama Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "Indiana@gmail.com",
   "765198390",
   "Indiana",
   "2019-11-23",
   "2 Elmwood St, Zionsville, INc 46077",
   "Indiana Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "Virginia@gmail.com",
   "4997319851",
   "Virginia",
   "1993-05-20",
   "47 Lake Forest St, Roanoke, VA 24012",
   "Virginia Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "Tiffin@gmail.com",
   "5568973421",
   "Tiffin",
   "2000-01-30",
   "221 Hickory St, Tiffin, OH 44883",
   "Tiffin Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "Clocks@gmail.com",
   "4109994322",
   "Clocks",
   "2006-07-20",
   “42 Clock Rd, Time Town, MI 48168 ",
   "Clocks Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "TV@gmail.com",
   "8235971094",
   "TV",
   "2010-12-15",
   "3 TV HQ Dr, Rolling Hills, OH 43201 ",
   "TV Store"
);
```

```
INSERT INTO SELLER_ACCOUNT
VALUES (
   "Lamp@gmail.com",
   "8879435674",
   "Lamp",
   "2019-03-12",
   "9999 Lamp Rd, Light City, CA 99430",
   "Lamp Store"
);


INSERT INTO SELLER_ACCOUNT
VALUES (
   "Tea@gmail.com",
   "4997834483",
   "Tea",
   "2004-05-17",
   "135 Tea Ct, Leafy Town, NC 55320",
   "Tea Store"
);


INSERT INTO SELLER_ACCOUNT
VALUES (
   "Computer@gmail.com",
   "4997302334",
   "Computer",
   "1999-09-05",
   "34 Computer Dr, Columbus, OH
43210",
   "Computer Store"
);


INSERT INTO SELLER_ACCOUNT
VALUES (
   "Bed@gmail.com",
   "4997304472",
   "Bed",
   "1950-03-10",
   "123 Mattress Rd, Bedding, CA 33201",
   "Bed Store"
);
```

```
INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "jontron@gmail.com",
    1
);

INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "thelegend27@gmail.com",
    2
);

INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "billship3@gmail.com",
    3
);

INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "LilySmith4@gmail.com",
    4
);
```

```
INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "PlaneJane5@gmail.com",
    5
);

INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "GeorgeLucas6@gmail.com",
    6
);

INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "Bubbawatson7@gmail.com",
    7
);

INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "BarackObama8@gmail.com",
    8
);
```

```
INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "MichelleObama9@gmail.com",
    9
);

INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "MattJones10@gmail.com",
    10
);

INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "AmongUs11@gmail.com",
    11
);

INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (
   "DrewToo12@gmail.com",
    12
);
```

```
INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (

   "GeneSith13@gmail.com",

    13

);


INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (

   "BooneJenner14@gmail.com",

    14

);


INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (

   "BluesClues15@gmail.com",

    15

);


INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (

   "GeorgeWashington16@gmail.com",

    16

);


INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (

   "harlesButt17@gmail.com",

    17

);


INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (

   "CharlesJones18@gmail.com",

    18

);


INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (

   "ChrisMerit19@gmail.com",

    19

);


INSERT INTO
BUYER_CONTAINS_TRANSACTION_R
ECORDS VALUES (

   "CharlotteWeb20@gmail.com",

    20

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "thelegend27@gmail.com",

    1

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "thelegend27@gmail.com",

    2

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "billship3@gmail.com",

    3

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "LilySmith4@gmail.com",

    4

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "PlaneJane5@gmail.com",

    5

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "GeorgeLucas6@gmail.com",

    6

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "Bubbawatson7@gmail.com",

    7

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "BarackObama8@gmail.com",

    8

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "MichelleObama9@gmail.com",

    9

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "MattJones10@gmail.com",

    10

);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "AmongUs11@gmail.com",

    11

);


IINSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(

   "DrewToo12@gmail.com",
```

```
    12
);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(
   "GeneSith13@gmail.com",
    13
);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(
   "BooneJenner14@gmail.com",
    14
);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(
   "BluesClues15@gmail.com",
    15
);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(v
   "GeorgeWashington16@gmail.com",
    16
);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(
   "harlesButt17@gmail.com",
    17
);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(
   "CharlesJones18@gmail.com",
    18
);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(
   "ChrisMerit19@gmail.com",
    19
);


INSERT INTO
BUYER_MANAGES_WISHLIST VALUES
(
   "CharlotteWeb20@gmail.com",
    20
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   1,
   "2020-10-23",
   "Google Store"
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   2,
   "2003-01-12",
   "Tiffin Store"
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   3,
   "2004-12-23",
   "Youtube Store"
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   4,
   "2007-03-03",
   "Google Store"
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   5,
   "2014-04-26",
   "New York Store"
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   6,
   "2016-11-04",
   "Ohio Store"
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   7,
   "2018-05-03",
   "Virginia Store"
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   8,
   "2006-01-28",
   "TV Store"
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   9,
   "2019-10-20",
   "Computer Store"
);


INSERT INTO TRANSACTION_RECORD
VALUES (
   10,
   "2020-10-13",
   "Clocks Store"
);
```

```
INSERT INTO TRANSACTION_RECORD VALUES (
  11,
  "2020-06-12",
  "Gmail Store"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  12,
  "2020-01-05",
  "Indiana Store"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  13,
  "2020-08-21",
  "Texas Store"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  14,
  "2020-08-11",
  "Alabama Store"
);

INSERT INTO WISHLIST VALUES (
  1,
  "2010-02-22"
);

INSERT INTO WISHLIST VALUES (
  2,
  "2020-05-21"
);

INSERT INTO WISHLIST VALUES (
  3,
  "2003-01-29"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  2,
  "2020-09-17",
  "Bed Store"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  15,
  "2020-12-19",
  "Google Store 2"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  16,
  "2020-09-23",
  "Google Store 3"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  17,
  "2020-02-21",
  "Lamp Store"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  18,
  "2020-05-14",
  "North Carolina Store"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  19,
  "2020-01-23",
  "Gmail Store"
);

INSERT INTO TRANSACTION_RECORD VALUES (
  20,
  "2020-04-12",
  "Tea Store"
);

INSERT INTO WISHLIST VALUES (
  4,
  "2020-06-18"
);

INSERT INTO WISHLIST VALUES (
  5,
  "2015-03-13"
);

INSERT INTO WISHLIST VALUES (
  6,
  "2020-08-27"
);

INSERT INTO WISHLIST VALUES (
  7,
  "2020-08-28"
);

INSERT INTO WISHLIST VALUES (
  8,
  "2020-09-24"
);
```

```
INSERT INTO WISHLIST VALUES (
   9,
   "2020-10-28"
);

INSERT INTO WISHLIST VALUES (
   10,
   "2020-02-16"
);

INSERT INTO WISHLIST VALUES (
   11,
   "2020-04-03"
);

INSERT INTO WISHLIST VALUES (
   12,
   "2020-09-05"
);

INSERT INTO WISHLIST VALUES (
   13,
   "2010-03-30"
);

INSERT INTO WISHLIST VALUES (
   14,
   "2020-02-29"
);

INSERT INTO WISHLIST VALUES (
   15,
   "2019-08-28"
);

INSERT INTO WISHLIST VALUES (
   16,
   "2018-08-28"
);

INSERT INTO WISHLIST VALUES (
   17,
   "2017-08-28"
);

INSERT INTO WISHLIST VALUES (
   18,
   "2020-07-28"
);

INSERT INTO WISHLIST VALUES (
   19,
   "2013-04-20"
);

INSERT INTO WISHLIST VALUES (
   20,
   "2020-03-21"
);

INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(
   "mike.123@bnb.com",
   1
);

INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(
   "Matt.2@bnb.com",
   2
);

INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(
   "Guess.3@bnb.com",
   3
);

INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(
   "Kylie.4@bnb.com",
   4
);

INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(
   "Taylor.5@bnb.com",
   5
);

INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(
   "Dan.6@bnb.com",
   6
);

INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(
   "Olivia.7@bnb.com",
   7
);

INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(
   "Jimmy.8@bnb.com",
```

```
    8

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "mike.123@bnb.com",

     11

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Ava.11@bnb.com",

     12

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Sophia.12@bnb.com",

     13

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Liam.9@bnb.com",

    "9

);
```

```
    "William.13@bnb.com",

     14

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Oliver.14@bnb.com",

     15

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Asher.15@bnb.com",

     16

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Brutus.16@bnb.com",

     17

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Emma.10@bnb.com",

    10

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Justin.17@bnb.com",

     18

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Luke.18@bnb.com",

     19

);
```

```
INSERT INTO
SPT_ACCT_CREATES_TICKET VALUES
(

    "Julius.19@bnb.com",

     20

);
```

```
INSERT INTO TROUBLE_TICKET
VALUES (

    1,

    "2020-10-20",

    "2020-10-23",

"This buyer was super annoying. I tried working with him but he was just super rude so I ended our support call abruptly."

);
```

```
INSERT INTO TROUBLE_TICKET
VALUES (

    2,

    "2020-01-12",

    "2020-01-15",

""What can I say about the 571B Banana Slicer that hasn't already been said about the wheel, penicillin, or the iPhone?"."

);
```

```
INSERT INTO TROUBLE_TICKET
VALUES (

    3,

    "2020-05-20",

    "2020-05-26",

"Gone are the days of biting off slice-sized chunks of banana and spitting them onto a serving tray.... Next on my wish list: a kitchen tool for dividing frozen water into cube-sized chunks."

);
```

```
INSERT INTO TROUBLE_TICKET
VALUES (
   4,
   "2020-03-25",
   "2020-03-26",
"As shown in the picture, the slices is curved from left to right. All of my bananas are bent the other way.”
);

INSERT INTO TROUBLE_TICKET
VALUES (
   5,
   "2019-10-20",
   "2019-10-23",
"I don't use it for vulgar endeavors like math or filling out a voter application, but BIC Cristal for Her is a lovely little writing utensil all the same. Ask your husband for some extra pocket money so you can buy one today!"
);

INSERT INTO TROUBLE_TICKET
VALUES (
   6,
   "2020-05-10",
   "2020-05-13",
"It was only after it arrived that I looked closely at the title and realized it said 'How to Avoid Huge SHIPS'. A simple error that means I am still treading on massive examples of canine excrement."
);

INSERT INTO TROUBLE_TICKET
VALUES (
   7,
   "2020-06-11",
   "2020-06-13",
"I read this book before going on vacation and I couldn't find my cruise liner in the port. Vacation ruined."
);

INSERT INTO TROUBLE_TICKET
VALUES (
   8,
   "2020-03-01",
   "2020-03-03",
"The cable knew where to go, and hooked itself into the correct ports without help from me.”
);

INSERT INTO TROUBLE_TICKET
VALUES (
   9,
   "2020-09-21",
   "2020-09-25",
"ust holding the packaging it comes in, I can see distant galaxies and, though you may not believe it, hear what the aliens there are thinking."
);

INSERT INTO TROUBLE_TICKET
VALUES (
   10,
   "2020-05-10",
   "2020-05-23",
"I purchased this product 4.47 Billion Years ago and when I opened it today, it was half empty."
);

INSERT INTO TROUBLE_TICKET
VALUES (
   11,
   "2020-10-21",
   "2020-10-24",
"I was very disappointed to have my uranium confiscated at the airport. It was a gift for my son for his birthday. Also, I'm in prison now, so that's not good either."
);

INSERT INTO TROUBLE_TICKET
VALUES (
   12,
   "2020-10-12",
   "2020-10-23",
"It is not cat food.... The cat's huge and well, doesn't really look much like a cat anymore."
);

INSERT INTO TROUBLE_TICKET
VALUES (
   13,
   "2020-10-20",
   "2020-10-30",
"They really need to put a warning label on this thing. Apparently, if you put it into your body, it turns into urine. Urine!"
);

INSERT INTO TROUBLE_TICKET
VALUES (
   14,
   "2020-10-10",
   "2020-10-11",
"Do you have any idea where this stuff comes from? It's excreted by squeezing the wobbly thingie on the UNDERSIDE OF A COW! That's hardly made clear anywhere on the label.."
);

INSERT INTO TROUBLE_TICKET
VALUES (
   15,
   "2020-10-20",
   "2020-10-20",
```

"Has anyone else tried pouring this stuff over dry cereal? A-W-E-S-O-M-E!"

);

INSERT INTO TROUBLE_TICKET VALUES (

16,

"2020-02-20",

"2020-02-23",

"t's OK Iguess, but the bumpy road majkes it hard to type. And theree's a lot of pedeestrians and traffic that keep distracting me fromm my computer.”

);

INSERT INTO TROUBLE_TICKET VALUES (

17,

"2020-04-20",

"2020-04-23",

"I love emailing the Highway patrol while I drive to let them know the tag numbers of cell phone using drivers."

);

INSERT INTO TROUBLE_TICKET VALUES (

18,

"2020-04-10",

"2020-04-13",

"I'm using it right now to post this review and I never”

);

INSERT INTO TROUBLE_TICKET VALUES (

19,

"2020-06-10",

"2020-06-12",

"HULK SAD. HULK DEMAND BIC FOR HIM."

);

INSERT INTO TROUBLE_TICKET VALUES (

20,

"2020-10-23",

"2020-10-23",

""This product is fantastic for those days when my prose is suffering from that not-so-fresh feeling."

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

"Google Store",

"google@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

"Youtube Store",

"youtuber@gmail.com”

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

“Florida Store",

"Florida@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

"New York Store",

"New York@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

"North Carolina Store",

"NorthCarolina@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

"Texas Store",

"Texas@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

"Google Store 2",

"google2@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

"Google Store 3",

"google3@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

"Gmail Store",

"Gmail@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

"Ohio Store",

"Ohio@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV VALUES (

```
  "Alabama Store",
 "Alabama@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV
VALUES (
   "Indiana Store",
"Indiana@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV
VALUES (
 "Virginia Store",
 "Virginia@gmail.com"
);

INSERT INTO SELL_ACCT_HOSTS_INV
VALUES (

INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Google Store",
   "Google strives to provide the best
   free products and outcompete
   microsoft for office products.",
   "The Google Store comprises of all
   Google products like gmail or docs
   which now all cost thousands of
   dollars.",
   "google.com"
);

INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 2",
   "Store Bio 2",
   "Store Descriptio 2",
   “Store2.com"
);

  “Tiffin Store",
 "Tiffin@gmail.com”
);

INSERT INTO SELL_ACCT_HOSTS_INV
VALUES (
 "Clocks Store",
 "Clocks@gmail.com"
);

INSERT INTO SELL_ACCT_HOSTS_INV
VALUES (
  "TV Store",
 "TV@gmail.com"

);

INSERT INTO SELL_ACCT_HOSTS_INV
VALUES (
   "Lamp Store",

INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 3",
   "Store Bio 3",
   "Store Descriptio 3",
   “Store3.com"
);

INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 4",
   "Store Bio 4",
   "Store Descriptio 4",
   “Store4.com"
);

INSERT INTO
VIRTUAL_INVENTORY VALUES (

 "Lamp@gmail.com"
);

INSERT INTO SELL_ACCT_HOSTS_INV
VALUES (
  "Tea Store",
"Tea@gmail.com"
);
INSERT INTO SELL_ACCT_HOSTS_INV
VALUES (
    "Computer Store",
"Computer@gmail.com"
);

INSERT INTO SELL_ACCT_HOSTS_INV
VALUES (
 "Bed Store",
 "Bed@gmail.com"
);

   "Store 5",
   "Store Bio 5",
   "Store Descriptio 5",
   “Store5.com"
);

INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 6",
   "Store Bio 6",
   "Store Descriptio 6",
   “Store6.com"
);

INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 7",
   "Store Bio 7",
   "Store Descriptio 7",
```

```
   “Store7.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 8",
    "Store Bio 8",
    "Store Descriptio 8",
   “Store8.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 9",
    "Store Bio 9",
    "Store Descriptio 9",
   “Store9.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 10",
    "Store Bio 10",
    "Store Descriptio 10",
   “Store10.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 11",
    "Store Bio 11",
    "Store Descriptio 11",
   “Store11.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 12",
    "Store Bio 12",
    "Store Descriptio 12",
   “Store12.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 13",
    "Store Bio 13",
    "Store Descriptio 13",
   “Store13.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 14",
    "Store Bio 14",
    "Store Descriptio 14",
   “Store14.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 15",
    "Store Bio 15",
    "Store Descriptio 15",
   “Store15.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 16",
    "Store Bio 16",
    "Store Descriptio 16",
   “Store16.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 17",
    "Store Bio 17",
    "Store Descriptio 17",
   “Store17.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 18",
    "Store Bio 18",
    "Store Descriptio 18",
   “Store18.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 19",
    "Store Bio 19",
    "Store Descriptio 19",
   “Store19.com"
);


INSERT INTO
VIRTUAL_INVENTORY VALUES (
   "Store 20",
    "Store Bio 20",
    "Store Descriptio 20",
   “Store20.com"
);
```

```
INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Microsoft Word",

   "Karma Points"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 2",

   "Credit Card"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 3",

   "Crypto Currency"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 4",

   "Karma Points"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 5",

   "Credit Card"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 6",

   "Crypto Currency"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 7",

   "Credit Card"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 8",

   "Crypto Currency"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 9",

   "Karma Points"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 10",

   "Credit Card"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 11",

   "Crypto Currency"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 12",

   "Karma Points"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 13",

   "Credit Card"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 14",

   "Crypto Currency"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 15",

   "Karma Points"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 16",

   "Credit Card"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 17",

   "Crypto Currency"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 18",

   "Karma Points"

);INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 19",

   "Credit Card"

);

INSERT INTO
ITEM_CANUSE_PAYMENT_TYPE
VALUES (

   "Item 20",

   "Crypto Currency"

);
```

```
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  1,
  "Microsoft Word"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  2,
  “Item 2"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  3,
  “Item 3"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  4,
  “Item 4"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  5,
  “Item 5"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  6,
  “Item 6"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  7,
  “Item 7"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  8,
  “Item 8"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  9,
  “Item 9"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  10,
  “Item 10"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  11,
  “Item 11"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  12,
  “Item 12"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  13,
  “Item 13"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  14,
  “Item 14"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  15,
  “Item 15"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  16,
  “Item 16"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  17,
  “Item 17"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  18,
  “Item 18"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  19,
  “Item 19"
);
INSERT INTO TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS VALUES (
  20,
  “Item 20"
);
```

```
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  1,
  "Microsoft Word"
);

INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  2,
  “Item 2"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  3,
  “Item 3"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  4,
  “Item 4"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  5,
  “Item 5"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  6,
  “Item 6"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  7,
  “Item 7"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES(
  8,
  “Item 8"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  9,
  “Item 9"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES(
  10,
  “Item 10"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  11,
  “Item 11"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  12,
  “Item 12"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  13,
  “Item 13"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  14,
  “Item 14"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  15,
  “Item 15"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  16,
  “Item 16"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES(
  17,
  “Item 17"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES(
  18,
  “Item 18"
);
INSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  19,
  “Item 19"
);
IINSERT INTO
WISHLIST_CONTAINS_IP_ITEM
VALUES (
  20,
  “Item 20"
);

INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
```

```
    1,
    "Microsoft Word"
);

INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES(
    2,
    “Item 2"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES(
    3,
    “Item 3"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    4,
    “Item 4"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    5,
    “Item 5"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES(
    6,
    “Item 6"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    7,
    “Item 7"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES(
    8,
    “Item 8"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES(
    9,
    “Item 9"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    10,
    “Item 10"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES(
    11,
    “Item 11"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    12,
    “Item 12"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    13,
    “Item 13"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    14,
    “Item 14"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    15,
    “Item 15"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES(
    16,
    “Item 16"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    17,
    “Item 17"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    18,
    “Item 18"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES (
    19,
    “Item 19"
);
INSERT INTO
INV_CONSISTSOF_IP_ITEMS
VALUES(
    20,
    “Item 20"
);


INSERT INTO PAYMENT_TYPE
VALUES (
    "Credit Card",
    "A virtual transaction consisting of
a credit card number, an expiration
date, and CCV."
);
```

```
INSERT INTO PAYMENT_TYPE
VALUES (
   "Cryptocurrency",
   "Using completely virtual stock
currencies such as Bitcoin or BAT"
);

INSERT INTO PAYMENT_TYPE
VALUES (
   "Karma Points",
   "Karma Points accumulated from
many purchases on this platform can
be applied to future purchases."
);

INSERT INTO IP_ITEM VALUES (
   "Microsoft Word",
   "document",
   "exe",
   "A program to help you write
papers for your english classes!",
   40.22
);

INSERT INTO IP_ITEM VALUES (
   "Item 2",
   "Keyword 2",
   "exe",
   "Decription 2",
   34.22
);
INSERT INTO IP_ITEM VALUES (
   "Item 3",
   "Keyword 3",
   "exe",
   "Decription 3",
   10.24
);
INSERT INTO IP_ITEM VALUES (
   "Item 4",
   "Keyword 4",
   "exe",
   "Decription 4",
   63.97
);
INSERT INTO IP_ITEM VALUES (
   "Item 5",
   "Keyword 5",
   "exe",
   "Decription 5",
   20.01
);
INSERT INTO IP_ITEM VALUES (
   "Item 6",
   "Keyword 6",
   "exe",
   "Decription 6",
   27.90
);
INSERT INTO IP_ITEM VALUES (
   "Item 7",
   "Keyword 7",
   "exe",
   "Decription 7",
   123.22
);
INSERT INTO IP_ITEM VALUES (
   "Item 8",
   "Keyword 8",
   "exe",
   "Decription 8",
   50.34
);
INSERT INTO IP_ITEM VALUES (
   "Item 9",
   "Keyword 9",
   "exe",
   "Decription 9",
   22.26
);
INSERT INTO IP_ITEM VALUES (
   "Item 10",
   "Keyword 10",
   "exe",
   "Decription 10",
   10.22
);
INSERT INTO IP_ITEM VALUES (
   "Item 11",
   "Keyword 11",
   "PDF",
   "Decription 11",
   10.22
);
INSERT INTO IP_ITEM VALUES (
   "Item 12",
   "Keyword 12",
   "PDF",
   "Decription 12",
   14.25
);
INSERT INTO IP_ITEM VALUES (
   "Item 13",
   "Keyword 13",
   "PDF",
   "Decription 13",
```

```
    10.78
);
INSERT INTO IP_ITEM VALUES (
    "Item 14",
    "Keyword 14”,
    "PDF",
    "Decription 14",
    15.93
);
INSERT INTO IP_ITEM VALUES (
    "Item 15",
    "Keyword 15",
    "PDF",
    "Decription 15",
    23.22
);
INSERT INTO IP_ITEM VALUES (
    "Item 16",
    "Keyword 16",
    "PDF",
    "Decription 16",
    10.22
);
INSERT INTO IP_ITEM VALUES (
    "Item 17",
    "Keyword 17",
    "PDF",
    "Decription 17",
    10.22
);
INSERT INTO IP_ITEM VALUES (
    "Item 18",
    "Keyword 18",
    "PDF",
    "Decription 18",
    10.22
);
INSERT INTO IP_ITEM VALUES (
    "Item 19",
    "Keyword 19",
    "PDF",
    "Decription 19",
    10.22
);
INSERT INTO IP_ITEM VALUES (
    "Item 20",
    "Keyword 20",
    "PDF",
    "Decription 20",
    10.22
);


INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Microsoft Word",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 2",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 3",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 4",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 5",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 6",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 7",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 8",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 9",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 10",
    "exe"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 11",
    "PDF"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 12",
    "PDF"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 13",
```

```
    "PDF"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 14",
    "PDF"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 15",
    "PDF"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 16",
    "PDF"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 17",
    "PDF"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 18",
    "PDF"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 19",
    "PDF"
);
INSERT INTO
IP_ITEM_HAS_IP_TYPE VALUES (
    "Item 20",
    "PDF"
);
```

```
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    1,
    "Credit Card"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    2,
    "Karma Points"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    3,
    "Crypto Currency"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    4,
    "Credit Card"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    5,
    "Karma Points"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    6,
    "Crypto Currency"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    7,
    "Credit Card"
);
```

```
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    8,
    "Karma Points"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    9,
    "Crypto Currency"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    10,
    "Credit Card"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    12,
    "Karma Points"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
   13,
    "Crypto Currency"
);INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    11,
    "Credit Card"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    14,
    "Karma Points"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
```

```
    15,
    "Crypto Currency"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    16,
    "Credit Card"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    17,
    "Karma Points"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    18,
    "Crypto Currency"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    19,
    "Credit Card"
);
INSERT INTO
TRANSACTION_RECORD_HAS_P
AYMENT_TYPE VALUES (
    20,
    "Karma Points"
);
```

```
INSERT INTO IP_TYPE VALUES (
    "This is an executable file.",
    "exe"
);
INSERT INTO IP_TYPE VALUES (
    "This is a PDF File.",
    "PDF"
);
INSERT INTO FEEDBACK VALUES
(
    1,
    "This product was very good. I
would buy this again.",
    5,
    "Microsoft Word"
);
INSERT INTO FEEDBACK VALUES
(
    2,
    "This product was very good. I
would buy this again.",
    4,
    "Item 2"
);
INSERT INTO FEEDBACK VALUES
(
    3,
    "This product was very good. I
would buy this again.",
    3,
    "Item 3"
);
INSERT INTO FEEDBACK VALUES
(
    4,
    "This product was very good. I
would buy this again.",
    5,
    "Item 4"
);
INSERT INTO FEEDBACK VALUES
(
    5,
    "This product was very good. I
would buy this again.",
    4,
    "Item 5"
);
INSERT INTO FEEDBACK VALUES
(
    6,
    "This product was very good. I
would buy this again.",
    4,
    "Item 6"
);
INSERT INTO FEEDBACK VALUES
(
    7,
    "This product was very terrible. I
would not buy this again.",
    1,
    "Item 7"
);
INSERT INTO FEEDBACK VALUES
(
    8,
    "This product was bad. I would not
buy this again.",
    2,
    "Item 8"
);
INSERT INTO FEEDBACK VALUES
(
    9,
    "This product was very good. I
would buy this again.",
    4,
    "Item 9"
```

```
);

INSERT INTO FEEDBACK VALUES
(

   10,

   "This product was very good. I
would buy this again.",

   4,

   "Item 10"

);

INSERT INTO FEEDBACK VALUES
(

   11,

   "This product was very bad. I
wouldn't buy this again.",

   1,

   "Item 11"

);
```

```
INSERT INTO FEEDBACK VALUES
(
   12,
   "This product was very good. I would buy this again.",
   5,
   "Item 12"
);
INSERT INTO FEEDBACK VALUES
(
   13,
   "This product was very bad. I wouldn’t buy this again.",
   1,
   "Item 13"
);
INSERT INTO FEEDBACK VALUES
(
   14,
   "Average",
   3,
   "Item 14"
);
INSERT INTO FEEDBACK VALUES
(
   15,
   "Close to average.",
   3,
   "Item 15"
);
INSERT INTO FEEDBACK VALUES
(
   16,
   "terrible",
   1,
   "Item 16"
);
INSERT INTO FEEDBACK VALUES
(
   17,
   "awful",
   1,
   "Item 17"
);
INSERT INTO FEEDBACK VALUES
(
   18,
   "Fantastic",
   5,
   "Item 18"
)
INSERT INTO FEEDBACK VALUES
(
   19,
   "YEs!",
   4,
   "Item 19"
);
INSERT INTO FEEDBACK VALUES
(
   20,
   "This product was very good. I would buy this again.",
   4,
   "Item 20"
);


INSERT INTO IMAGE VALUES (
"assets/logo_homepage.normal.v108.svg",
   "A picture of something. Not really sure what it is, actually."
);

INSERT INTO IMAGE VALUES (
"www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png",
   "The Google logo"
);
INSERT INTO IMAGE VALUES (
"www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png",
   "Item 3"
);
INSERT INTO IMAGE VALUES (
"www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png",
   "Item 4"
);
INSERT INTO IMAGE VALUES (
"www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png",
   "Item 5"
);
INSERT INTO IMAGE VALUES (
"www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png",
   "Item 6"
);
INSERT INTO IMAGE VALUES (
"www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png",
   "Item 7"
);
INSERT INTO IMAGE VALUES (
"www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png",
   "Item 8"
);
```

```
INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 9"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 10"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 11"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 12"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 13"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 14"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 15"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 16"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 17"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 18"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 19"

);

INSERT INTO IMAGE VALUES (

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png",

   "Item 20"

);



INSERT INTO INV_HAS_IMAGES
VALUES (

   "Google Store",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

   "Store 2",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

   "Store 3",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

   "Store 4",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

   "Store 2",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

   "Store 5",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (
```

```
    "Store 6",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 7",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 8",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 9",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 10",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 11",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 12",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 13",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 14",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 15",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 16",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 17",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 18",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 19",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);

INSERT INTO INV_HAS_IMAGES
VALUES (

    "Store 20",

"www.google.com/images/branding/
googlelogo/1x/googlelogo_color_272
x92dp.png"

);



INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

    "Microsoft Word",

"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

    "Item 2",

"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

    "Item 3",
```

```
"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 4",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 5",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 6",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 7",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 8",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 9",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 2",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 10",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 11",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 12",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 13",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 14",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 15",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 16",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 17",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 18",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 19",


"assets/logo_homepage.normal.v108
.svg"

);

INSERT INTO
IP_ITEM_HAS_IMAGES VALUES (

   "Item 20",


"assets/logo_homepage.normal.v108
.svg"

);
```

3. Given your relational schema, provide the SQL to perform the following queries. If your schema cannot provide answers to these queries, revise your ER Model and your relational schema to contain the appropriate information for these queries. These queries should be provided in a plain text file named "SimpleQueries.txt":

a. Find the titles of all IP Items by a given Seller that cost less than $10 (you choose how to designate the seller)

Let the given Seller be identified by the unique email 'smithybob@rocketmail.com'

```
SELECT Item_name
FROM IP_ITEM AS a, VIRTUAL_INVENTORY AS b, INV_CONSISTSOF_IP_ITEMS AS c, SELL_ACCT_HOSTS_INV AS d
WHERE d.Seller_email = 'smithybob@rocketmail.com'
AND d.Store_name = b.Store_name
AND b.Store_name = c.Store_name
AND c.Item_name = a.Item_name
AND a.Price < 10.00;
```

b. Give all the titles and their dates of purchase made by given buyer (you choose how to designate the buyer)

Let the given Buyer be identified by the unique email 'smithybob@rocketmail.com'

```
SELECT a.Item_name, a.Date
FROM TRANSACTION_RECORD AS a, BUYER_CONTAINS_TRANSACTION_RECORDS AS b
WHERE b.Buyer_email = 'smithybob@rocketmail.com'
AND b.Transaction_num = a.Transaction_num;
```

c. Find the seller names for all sellers with less than 5 IP Items for sale

```
SELECT a.Seller_email
FROM SELL_ACCT_HOSTS_INV AS a, VIRTUAL_INVENTORY AS b, IP_ITEM AS c,
INV_CONSISTSOF_IP_ITEMS AS d
WHERE Count(c.Item_name) < 5
AND c.Item_name = d.Item_name
AND d.Store_name = b.Store_name
AND b.Store_name = a.Store_name;
```

d. Give all the buyers who purchased a IP Item by a given seller and the names of the IP Items they purchased

Let the given Seller be identified by the unique email 'smithybob@rocketmail.com'

```
SELECT g.Buyer_email, c.Item_name
FROM SELL_ACCT_HOSTS_INV AS a, VIRTUAL_INVENTORY AS b,
INV_CONSISTSOF_IP_ITEMS AS c, IP_ITEM AS d,
TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS AS e, TRANSACTION_RECORDS AS f,
BUYER_CONTAINS_TRANSACTION_RECORDS AS g
WHERE a.Seller_email = 'smithybob@rocketmail.com'
AND a.Store_name = b.Store_name
AND b.Store_name = c.Store_name
AND c.Item_name = d.Item_name
AND d.Item_name = e.Item_name
AND e.Transaction_num = f.Transaction_num
AND f.Transaction_num = g.Transaction_num;
```

e. Find the total number of IP Items purchased by a single buyer (you choose how to designate the buyer)

Let the given Buyer be identified by the unique email 'smithybob@rocketmail.com'

```
SELECT Count(a.Item_name)
```

FROM TRANSACTION_RECORD AS a, BUYER_CONTAINS_TRANSACTION_RECORDS AS b

WHERE b.Buyer_email = 'smithybob@rocketmail.com'

AND b.Transaction_num = a.Transaction_num;

f. Find the buyer who has purchased the most IP Items and the total number of IP Items they have purchased

CREATE VIEW Buyers_And_Items_Bought(Buyer_email, Buyer_item_count)

AS

SELECT b.Buyer_email, Count(a.Transaction_num)

FROM TRANSACTION_RECORD AS a, BUYER_CONTAINS_TRANSACTION_RECORDS AS b

WHERE b.Transaction_num = a.Transaction_num;

SELECT *

FROM Buyers_And_Items_Bought

WHERE MAX(Buyer_item_count);

4. For Project Checkpoint 02, you were asked to come up with three additional interesting queries that your database can provide. Provide the SQL to perform those queries. Your queries should include at least one of these:

a. outer joins - get a listing of all buyer accounts and seller accounts

SELECT a.Email, b.Email

FROM SELLER_ACCOUNT AS a, BUYER_ACCOUNT AS b;

b. aggregate function - find a virtual store's most expensive listing

Let the given Virtual Inventory be identified by the unique name 'Google Store'

```
SELECT Item_name, MAX(b.Price)
FROM VIRUAL_INVENTORY AS a, IP_Item AS b
WHERE a. Store_name = 'Google Store'
AND a.Item_name = b.Item_name;
```

c. "extra" entities from CP01 - find all accounts that a support account has managed

Let the given Support Account be identified by the unique email 'smithybob@rocketmail.com'

```
SELECT Seller_email, Buyer_email
FROM SELLERS_MANAGED, BUYERS_MANAGED
WHERE Seller_email = 'smithybob@rocketmail.com'
OR Buyer_email = 'smithybob@rocketmail.com';
```

5. Given your relational schema, provide the SQL for the following more advanced queries. These queries may require you to use techniques such as nesting, aggregation using having clauses, and other SQL techniques.

a. Provide a list of buyer names, along with the total dollar amount each buyer has spent.

```
CREATE VIEW BUYERS_TOTALS(Buyer_name, Buyer_email, Transaction_total)
SELECT a.Name, a.Email, SUM(d.Price)
FROM BUYER_ACCOUNT AS a, BUYER_CONTAINS_TRANSACTION_RECORDS AS b,
TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS AS c, IP_ITEM AS d
WHERE a.Email = b.Buyer_email
AND b.Transaction_num = c.Transaction_num
AND c.item_name = d.Item_name
GROUP BY a.Email;

SELECT Buyer_name, Transaction_total
```

```
FROM BUYER_BUYERS_TOTALS;
```

b. Provide a list of buyer names and email addresses for buyers who have spent more than the average buyer.

```
SELECT Buyer_name, Buyer_email
FROM BUYER_BUYERS_TOTALS
WHERE Total_spendings > AVG(Total_spendings);
```

c. Provide a list of the IP Item names and associated total copies sold to all buyers, sorted from the IP Item that has sold the most individual copies to the IP Item that has sold the least.

```
CREATE VIEW ITEM_TRANSACTION_IDS
SELECT I.Item_name, R.Transaction_num
FROM IP_ITEM as I, TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS as R
WHERE I.Item_name = R.Item_name;

CREATE VIEW ITEM_TRANSACTIONS
SELECT I.Item_name, R.Transaction_num
FROM ITEM_TRANSACTION_IDS as I, TRANSACTION_RECORD as R
WHERE I.Transaction_num = R.Transaction_num;

CREATE VIEW IP_ITEM_SOLD_COPIES
SELECT Item_name, count(distinct Item_name)
FROM ITEM_TRANSACTIONS
ORDER BY 2 DESC;
```

d. Provide a list of the IP Item names and associated dollar totals for copies sold to all buyers, sorted from the IP Item that has sold the highest dollar amount to the IP Item that has sold the smallest.

```
CREATE VIEW ITEM_TRANSACTION_IDS
SELECT I.Item_name, I.Price, R.Transaction_num
FROM IP_ITEM as I, TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS as R
WHERE I.Item_name = R.Item_name;

CREATE VIEW ITEM_TRANSACTIONS
SELECT I.Item_name, I.Price, R.Transaction_num
FROM ITEM_TRANSACTION_IDS as I, TRANSACTION_RECORD as R
WHERE I.Transaction_num = R.Transaction_num;

SELECT Item_name, sum(Price)
FROM ITEM_TRANSACTIONS
GROUP BY Item_name;
```

e. Find the most popular Seller (i.e. the one who has sold the most IP Items)

```
CREATE VIEW STORE_ITEMS(Seller_Email, Item_count)
SELECT a.Seller_email, COUNT(b.Item_name)
FROM SELL_ACCT_HOSTS_INV AS a, VIRTUAL_INVENTORY AS b,
INV_CONSISTSOF_IP_ITEMS as c
WHERE a.Store_name = b.Store_name AND b.Store_name = c.Store_name;

CREATE VIEW SELLER_ITEMS(Seller_email, Total_items)
SELECT COUNT(b.Item_count)
FROM SELLER_ACCOUNT AS a, STORE_ITEMS AS b
WHERE a.Email = b.Seller_email;

SELECT a.Name
FROM SELLER_ACCOUNT AS a, SELLER_ITEMS AS b
WHERE MAX(b.Total_items)
```

AND b.Seller_email = a.Email;

f. Find the most profitable seller (i.e. the one who has brought in the most money)

CREATE VIEW STORE_REVENUE(Seller_email, Store_money)

SELECT a.Store_name, SUM(b.Price)

FROM SELL_ACCT_HOSTS_INV AS a, VIRTUAL_INVENTORY AS b,
INV_CONSISTSOF_IP_ITEMS as c

WHERE a.Store_name = b.Store_name AND b.Store_name = c.Store_name;

CREATE VIEW SELLER_REVENUE(Seller_email, Total_revenue)

SELECT SUM(b.Store_money)

FROM SELLER_ACCOUNT AS a, STORE_REVENUE AS b

WHERE a.Email = b.Seller_email;

SELECT a.Name

FROM SELLER_ACCOUNT AS a, SELLER_REVENUE AS b

WHERE MAX(b.Total_revenue)

AND b.Seller_email = a.Email;

g. Provide a list of buyer names for buyers who purchased anything listed by the most profitable Seller.

Let the given Seller be identified by the unique email 'smithybob@rocketmail.com' using the query above

CREATE VIEW SELLER_INV

SELECT VI.Store_name

FROM SELL_ACCT_HOSTS_INV AS SAHI, VIRTUAL_INVENTORY AS VI

WHERE SAHI.Store_name = VI.Store_name AND SAHI.Seller_email =
"smithybob@rocketmail.com";

CREATE VIEW INV_ITEMS

SELECT II.Item_name

FROM SELLER_INV AS SI, INV_CONSISTSOF_IP_ITEMS AS ICII, IP_ITEM as II

WHERE SI.Store_name = ICII.Store_name AND II.Item_name = ICII.Item_name

CREATE VIEW TRANSACTIONS

SELECT TR.Transaction_num

FROM INV_ITEMS AS II, TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS AS CO, TRANSACTION_RECORDS AS TR

WHERE CO.Item_name = II.Item_name AND CO.Transaction_num = TR.Transaction_num

SELECT B.Name

FROM TRANSACTIONS AS T, BUYER_ACCOUNT AS B, BUYER_CONTAINS_TRANSACTION_RECORDS AS BCTR

WHERE BCTR.Transaction_num = T.Transaction_num AND BCTR.Buyer_email = B.Email

h. Provide the list of sellers who listed the IP Items purchased by the buyers who have spent more than the average buyer.

CREATE VIEW BUYER_TRANSACTIONS

SELECT BCTR.Buyer_email, BCTR.Transaction_num, TR.Store_name

FROM TRANSACTION_RECORDS AS TR, BUYER_CONTAINS_TRANSACTION_RECORDS AS BCTR

WHERE BCTR.Transaction_num = TR.Transaction_num

CREATE VIEW TRANSACTION_ITEMS

SELECT BT.Buyer_email, BT.Store_num, II.Price

FROM BUYER_TRANSACTIONS AS BT, TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS AS CO, IP_ITEM AS II,

WHERE CO.Transaction_num = BT.Transaction_num AND CO.Item_name = II.Item_name

```
CREATE VIEW BUYER_TOTALS
SELECT Buyer_email, Store_num, sum(Price)
FROM TRANSACTION_ITEMS
GROUP BY Buyer_email

CREATE VIEW BUYERS_HIGH_SPENDING
SELECT Buyer_email, Store_num
FROM BUYER_TOTALS
WHERE Price > avg(price)

SELECT S.Name
FROM SELL_ACCT_HOSTS_INV AS SAHI, BUYERS_HIGH_SPENDING AS BHS,
SELLER_ACCOUNT AS S
WHERE SAHI.Store_name = BHS.Store_num AND SAHI.Seller_email = S.Email
```

CSE 3241 Project Checkpoint 04

Sam Bossley | Michael Izzo | Josephine Ko | Henry Xiong

COMMENTS

diagram looks goo dafter revisons from cp02 - queries and schema are also looking good. i do not have any additional suggestions - good job

1. Provide a current version of your ER Diagram and Relational Model as per Project Checkpoint 03. If you were instructed to change the model for Project Checkpoint 03, make sure you use the revised versions of your models.

ERD was left unchanged, some errors in the Relational Schema were fixed.

See Figure 1: Revised ERD v3.1 on page 75

Figure 1: ERD v3.0

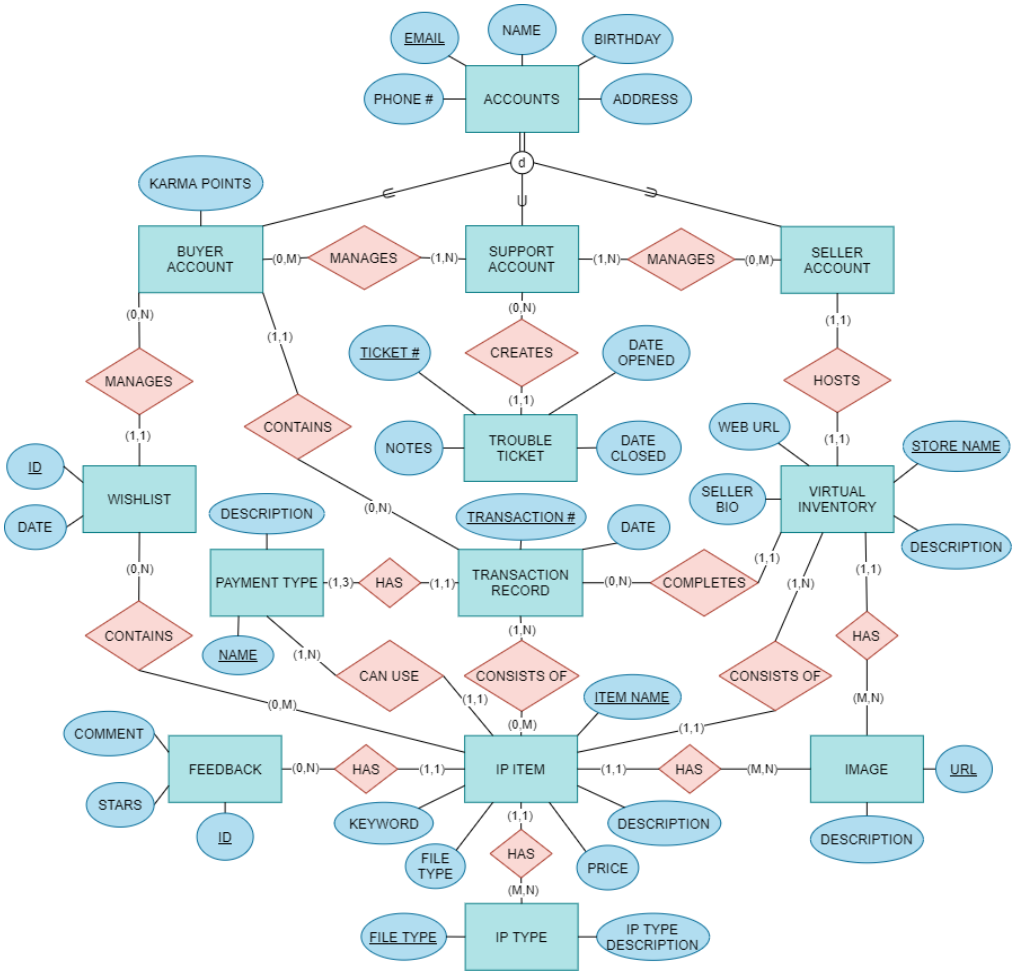
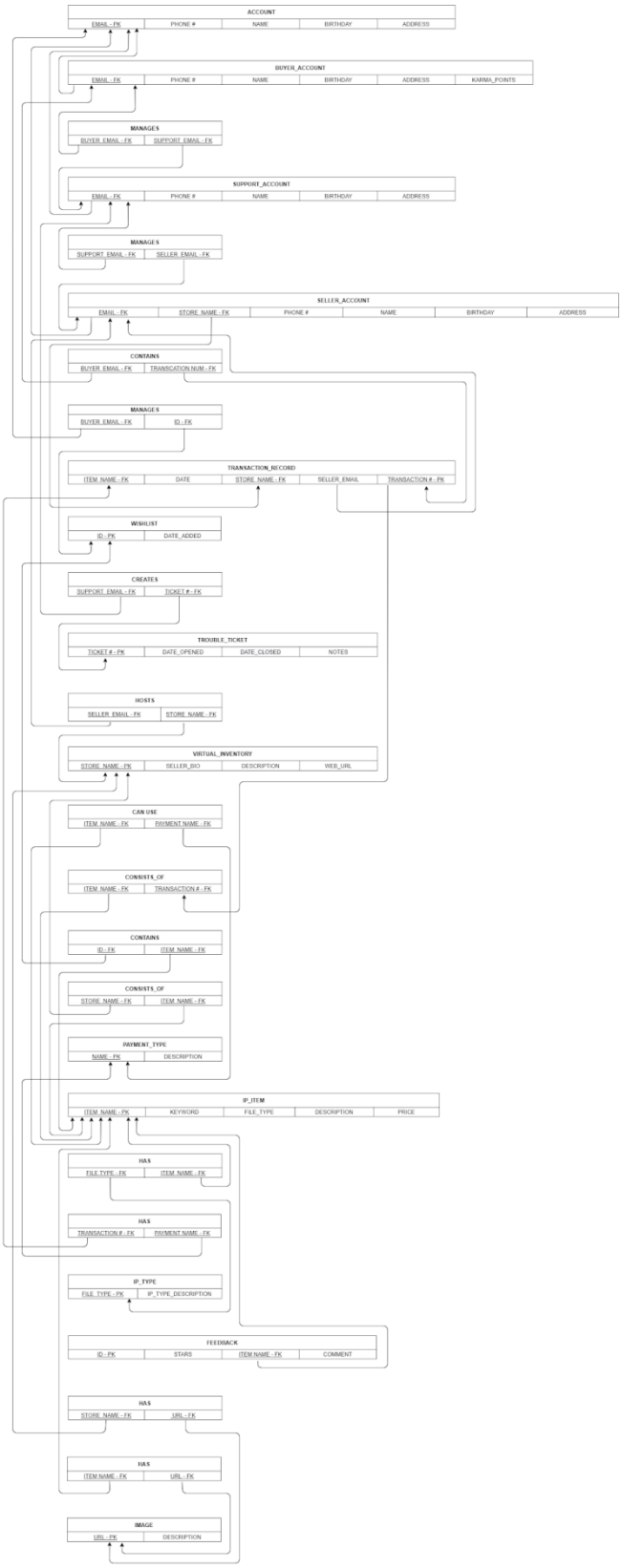


Figure 2: Relational Schema v1.1



2. For each relation schema in your model, indicate the functional dependencies.

Think carefully about what you are modeling here - make sure you consider all the possible dependencies in each relation and not just the ones from yours. For example, a customer's credit card number is unique and will uniquely identify a customer even if you have another key in the same table (in fact, if the customer can have multiple credit card numbers, the dependencies can get even more involved).

ACCOUNT

R = { Email, Phone_num, Name, Birthday, Address }

F = { Email → Phone_num, Name, Birthday, Address }

BUYER_ACCOUNT

R = { Email, Phone_num, Name, Birthday, Address }

F = { Email → Phone_num, Name, Birthday, Address }

SUPPORT_MANAGES_BUYER

R = { Buyer_email (FK), Support_email (FK) }

No dependencies

SUPPORT_ACCOUNT

R = {Email, Phone_num, Name, Birthday, Address }

F = {Email → Phone_num, Name, Birthday, Address }

SUPPORT_MANAGES_SELLER

R = { Support_email (FK), Seller_email (FK) }

No dependencies

SELLER_ACCOUNT

R = { Email, Phone_num, Name, Birthday, Address }

F = { Email → Phone_num, Name, Birthday, Address }

BUYER_CONTAINS_TRANSACTION_RECORDS

R = { Buyer_email (FK), Transaction_num (FK) }

No dependencies

BUYER_MANAGES_WISHLIST

R = { Buyer_email (FK), Id (FK) }

No dependencies

TRANSACTION_RECORD

R = { Transaction_num, Date, Store_name (FK), Item_name (FK) }

F = { Transaction_num → Date, Store_name (FK), Item_name (FK) }

WISHLIST

R = { Id, Date_added }

F = { Id → Date_added }

SPT_ACCT_CREATES_TICKET

R = { Support_email (FK), Ticket_num (FK) }

No dependencies

TROUBLE_TICKET

R = { Ticket_num, Date_opened, Date_closed, Notes }

F = { Ticket_num → Date_opened, Date_closed, Notes }

SELL_ACCT_HOSTS_INV

R = { Store_name (FK), Seller_email (FK) }

No dependencies

VIRTUAL_INVENTORY

R = { Store_name, Seller_bio, Description, Web_url }

F = { Store_name → Seller_bio, Description, Web_url }

ITEM_CANUSE_PAYMENT_TYPE

R = { Item_name (FK), Payment_name (FK) }

No dependencies

TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS

R = { Item_name (FK), Transaction_num (FK) }

No dependencies

WISHLIST_CONTAINS_IP_ITEM

R = { Id (FK), Item_name (FK) }

No dependencies

INV_CONSISTSOF_IP_ITEMS

R = { Store_name (FK), Item_name (FK) }

No dependencies

PAYMENT_TYPE

R = { Name, Description }

F = { Name → Description }

IP_ITEM

R = { Item_name, Keyword, File_type, Description, Price }

F = { Item_name → Keyword, File_type, Description, Price }

IP_ITEM_HAS_IP_TYPE

R = { File_type (FK), Item_name (FK) }

No dependencies

TRANSACTION_RECORD_HAS_PAYMENT_TYPE

R = { Transaction_num (FK), Payment_name (FK) }

No dependencies

IP_TYPE

R = { File_type, IP_type_description }

F = { File_type → IP_type_description }

FEEDBACK

R = { Id, Stars, Item_name (FK), Comment }

F = { Id → Stars, Item_name (FK), Comment }

INV_HAS_IMAGES

R = { Store_name (FK), Url (FK) }

No dependencies

IP_ITEM_HAS_IMAGES

R = { Item_name (FK), Url (FK) }

No dependencies

IMAGE

R = { Url, Description }

F = { Url → Description }

3. For each relation schema in your model, determine the highest normal form of the relation. If the relation is not in 3NF, rewrite your relation schema so that it is in at least 3NF.

See revision on page 76

ACCOUNT

R = { Email, Phone_num, Name, Birthday, Address }

3NF = { Email → Phone_num, Name, Birthday, Address }

BUYER_ACCOUNT

R = { Email, Phone_num, Name, Birthday, Address }

3NF = { Email → Phone_num, Name, Birthday, Address }

SUPPORT_MANAGES_BUYER

R = { Buyer_email (FK), Support_email (FK) }

No dependencies, 3NF

SUPPORT_ACCOUNT

R = {Email, Phone_num, Name, Birthday, Address }

3NF = {Email → Phone_num, Name, Birthday, Address }

SUPPORT_MANAGES_SELLER

R = { Support_email (FK), Seller_email (FK) }

No dependencies, 3NF

SELLER_ACCOUNT

R = { Email, Phone_num, Name, Birthday, Address }

3NF = { Email → Phone_num, Name, Birthday, Address }

BUYER_CONTAINS_TRANSACTION_RECORDS

R = { Buyer_email (FK), Transaction_num (FK) }

No dependencies, 3NF

BUYER_MANAGES_WISHLIST

R = { Buyer_email (FK), Id (FK) }

No dependencies, 3NF

TRANSACTION_RECORD

R= { Transaction_num, Date, Store_name (FK), Item_name (FK) }

1NF = { Transaction_num → Date, Store_name (FK), Item_name (FK) }

3NF = { Transaction_num, Store_name (FK), Item_name (FK) → Date }

WISHLIST

R = { Id, Date_added }

3NF = { Id → Date_added }

SPT_ACCT_CREATES_TICKET

R = { Support_email (FK), Ticket_num (FK) }

No dependencies, 3NF

TROUBLE_TICKET

R = { Ticket_num, Date_opened, Date_closed, Notes }

3NF = { Ticket_num → Date_opened, Date_closed, Notes }

SELL_ACCT_HOSTS_INV

R = { Store_name (FK), Seller_email (FK) }

No dependencies, 3NF

VIRTUAL_INVENTORY

R = { Store_name, Seller_bio, Description, Web_url }

3NF = { Store_name → Seller_bio, Description, Web_url }

ITEM_CANUSE_PAYMENT_TYPE

R = { Item_name (FK), Payment_name (FK) }

No dependencies, 3NF

TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS

R = { Item_name (FK), Transaction_num (FK) }

No dependencies, 3NF

WISHLIST_CONTAINS_IP_ITEM

R = { Id (FK), Item_name (FK) }

No dependencies, 3NF

INV_CONSISTSOF_IP_ITEMS

R = { Store_name (FK), Item_name (FK) }

No dependencies, 3NF

PAYMENT_TYPE

R = { Name, Description }

3NF = { Name → Description }

IP_ITEM

R = { Item_name, Keyword, File_type, Description, Price }

3NF = { Item_name → Keyword, File_type, Description, Price }

IP_ITEM_HAS_IP_TYPE

R = { File_type (FK), Item_name (FK) }

No dependencies, 3NF

TRANSACTION_RECORD_HAS_PAYMENT_TYPE

R = { Transaction_num (FK), Payment_name (FK) }

No dependencies, 3NF

IP_TYPE

R = { File_type, IP_type_description }

3NF = { File_type → IP_type_description }

FEEDBACK

R = { Id, Stars, Item_name (FK), Comment }

1NF = { Id → Stars, Item_name (FK), Comment }

3NF = { Id, Item_name (FK) → Comment, Stars }

INV_HAS_IMAGES

R = { Store_name (FK), Url (FK) }

No dependencies, 3NF

IP_ITEM_HAS_IMAGES

R = { Item_name (FK), Url (FK) }

No dependencies, 3NF

IMAGE

R = { Url, Description }

3NF = { Url → Description }

4. For each relation schema in your model that is in 3NF but not in BCNF, either rewrite the relation schema to BCNF or provide a short justification for why this relation should be an exception to the rule of putting relations into BCNF.

All relations in the relational schema are already in BCNF.

5. For your database, propose at least two interesting views that can be built from your relations. These views must involve joining at least two tables together each and must include some kind of aggregation in the view. Each view must also be able to be described by a one or two sentence description in plain English. Provide

the code for constructing your views along with the English language description of what the view is supposed to be providing.

1. Finding a virtual store's most expensive listing (where VIRTUAL_INVENTORY is identified by the unique name 'Google Store'):

   CREATE VIEW STORE_MOST_EXPENSIVE

   SELECT Item_name, MAX(b.Price)

   FROM VIRUAL_INVENTORY AS a, IP_Item AS b

   WHERE a. Store_name = 'Google Store'

   AND a.Item_name = b.Item_name;

2. Finding all accounts that a specific support account has managed (where SUPPORT_ACCOUNT is identified by the unique email 'smithybob@rocketmail.com'):

   CREATE VIEW ALL_MANAGED_ACCOUNTS

   SELECT Seller_email, Buyer_email

   FROM SUPPORT_MANAGES_SELLER as S, SUPPORT_MANAGES_BUYER as B

   WHERE B.Support_email = 'smithybob@rocketmail.com'

   OR S.Support_email = 'smithybob@rocketmail.com';

6. Description of two indexes that you want to implement in your DB. Explain their purpose and what you want to achieve by implementing them. Explain what type of indexing would be most appropriate for each one of them (Clustering, Hash, or B-tree) and why.

1. The first index is indexing for a specific store name. This is helpful because users of the database can find the exact store name they are looking for quickly. Hashing would be the most appropriate type of indexing because it is the quickest in regard to using equalities.

2. The second index is indexing for an IP item with specifications on the cost range. This is helpful because users can identify certain items within the range of cost they have determined they want to spend. The best type of indexing for this would be B-Tree, this is the fastest way to find all of the values the user is asking for.

7. Two sample transactions that you want to establish in your DB. Clearly document their purpose and function. Include the sample SQL code for each transaction. Each transaction should include read and/or write operations on at least two tables, with appropriate error and constraint checks and responses.

1. Removes an IP_ITEM from VIRTUAL_INVENTORY (Where VIRTUAL_INVENTORY is uniquely identified by the store name ‘Google Store’ and IP_ITEM is uniquely identified by the name ‘Virus.exe’)

```
BEGIN TRANSACTION REMOVE_IP_ITEM

    CREATE VIEW ALL_IP_ITEMS
    SELECT Store_name, Item_name
    FROM VIRTUAL_INVENTORY AS V
        INV_CONSISTSOF_IP_ITEMS AS C,
        IP_ITEM AS I
    WHERE V.Store_name = ‘Google Store’
          AND V.Store_name = C.Store_name
          AND C.Item_name = I.Item_name

    IF error THEN GO TO UNDO; END IF;

        DELETE FROM IP_ITEM
        WHERE I.Item_name ='Virus.exe';
        IF error THEN GO TO UNDO; END IF;

        COMMIT;
        GO TO FINISH;
    UNDO:
        ROLLBACK;
    FINISH:
```

END TRANSACTION;

2. A BUYER_ACCOUNT adds an IP_ITEM to their WISHLIST (Where BUYER_ACCOUNT is uniquely identified by email ‘smithybob@rocketmail.com’ and IP_ITEM is uniquely identified by the name ‘Virus.exe’):

```
BEGIN TRANSACTION WISHLIST_ADD_ITEM

    CREATE VIEW WISHLIST_TOTAL_LIST
    SELECT Buyer_email, Item_name
    FROM BUYER_MANAGES_WISHLIST AS M,
        WISHLIST_CONTAINS_IP_ITEM AS W
    WHERE M.Id = W.Id
        AND M.Buyer_email = ‘smithybob@rocketmail.com’;

        IF error THEN GO TO UNDO; END IF;

        INSERT INTO WISHLIST_TOTAL_LIST VALUES (
            ‘smithybob@rocketmail.com’,
            ‘Virus.exe’
        );

        IF error THEN GO TO UNDO; END IF;

    COMMIT;
    GO TO FINISH;
    UNDO:
        ROLLBACK;
    FINISH:
END TRANSACTION;
```

CSE 3241 Project Checkpoint 04 Revisions

Sam Bossley | Michael Izzo | Josephine Ko | Henry Xiong

COMMENTS

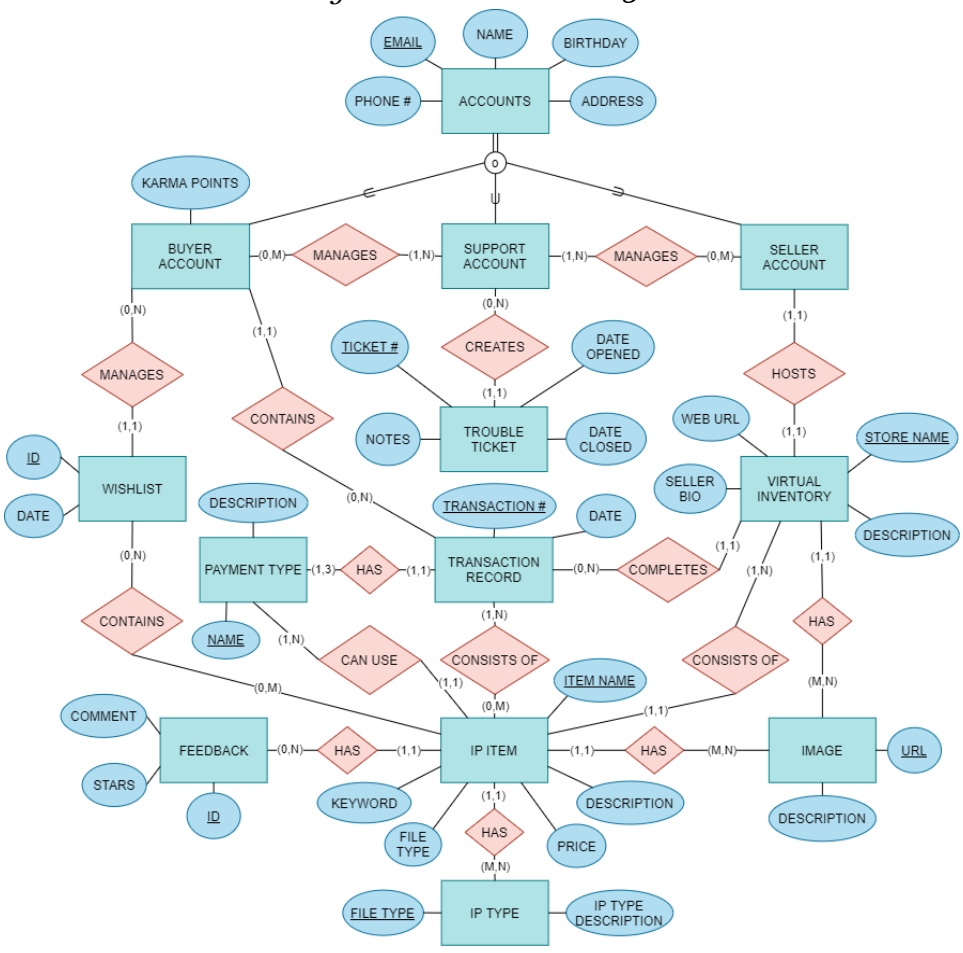
accounts -> buyer/seller should be overlap rather than disjoint. I thought I mentioned this before but I might have not.

3. Each table should be in 3NF, accompanied by a meaningful and specific rationalization for each table. For example, -
“the PK is atomic, so the table is in 2nf”
-“all of the FDs have the whole PK as the determinant, so the table is in BCNF”
-“the table is binary, and so in BCNF”

1. Provide a current version of your ER Diagram and Relational Model as per Project Checkpoint 03.

ERD corrected: disjoint → overlap

Figure 1: Revised ERD v3.1



3. For each relation schema in your model, determine the highest normal form of the relation. If the relation is not in 3NF, rewrite your relation schema so that it is in at least 3NF.

ACCOUNT
R = { Email, Phone_num, Name, Birthday, Address }
3NF (BCNF) = { Email → Phone_num, Name, Birthday, Address }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

BUYER_ACCOUNT
R = { Email, Phone_num, Name, Birthday, Address }
3NF (BCNF) = { Email → Phone_num, Name, Birthday, Address }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

SUPPORT_MANAGES_BUYER
R = { Buyer_email (FK), Support_email (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

SUPPORT_ACCOUNT
R = {Email, Phone_num, Name, Birthday, Address }
3NF (BCNF) = {Email → Phone_num, Name, Birthday, Address }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

SUPPORT_MANAGES_SELLER
R = { Support_email (FK), Seller_email (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

SELLER_ACCOUNT
R = { Email, Phone_num, Name, Birthday, Address }
3NF (BCNF) = { Email → Phone_num, Name, Birthday, Address }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

BUYER_CONTAINS_TRANSACTION_RECORDS
R = { Buyer_email (FK), Transaction_num (FK) }
No dependencies, 3NF (BCNF)

This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

BUYER_MANAGES_WISHLIST
R = { Buyer_email (FK), Id (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

TRANSACTION_RECORD
R= { Transaction_num, Date, Store_name (FK), Item_name (FK) }
1NF = { Transaction_num → Date, Store_name (FK), Item_name (FK) }
3NF (BCNF) = { Transaction_num, Store_name (FK), Item_name (FK) → Date }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

WISHLIST
R = { Id, Date_added }
3NF (BCNF) = { Id → Date_added }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

SPT_ACCT_CREATES_TICKET
R = { Support_email (FK), Ticket_num (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

TROUBLE_TICKET
R = { Ticket_num, Date_opened, Date_closed, Notes }
3NF (BCNF) = { Ticket_num → Date_opened, Date_closed, Notes }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

SELL_ACCT_HOSTS_INV
R = { Store_name (FK), Seller_email (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

VIRTUAL_INVENTORY
R = { Store_name, Seller_bio, Description, Web_url }
3NF (BCNF) = { Store_name → Seller_bio, Description, Web_url }

This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

ITEM_CANUSE_PAYMENT_TYPE
R = { Item_name (FK), Payment_name (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

TRANSACTION_RECORDS_CONSISTOF_IP_ITEMS
R = { Item_name (FK), Transaction_num (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

WISHLIST_CONTAINS_IP_ITEM
R = { Id (FK), Item_name (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

INV_CONSISTSOF_IP_ITEMS
R = { Store_name (FK), Item_name (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

PAYMENT_TYPE
R = { Name, Description }
3NF (BCNF) = { Name → Description }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

IP_ITEM
R = { Item_name, Keyword, File_type, Description, Price }
3NF (BCNF) = { Item_name → Keyword, File_type, Description, Price }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

IP_ITEM_HAS_IP_TYPE
R = { File_type (FK), Item_name (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

TRANSACTION_RECORD_HAS_PAYMENT_TYPE
R = { Transaction_num (FK), Payment_name (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

IP_TYPE
R = { File_type, IP_type_description }
3NF (BCNF) = { File_type → IP_type_description }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

FEEDBACK
R = { Id, Stars, Item_name (FK), Comment }
1NF = { Id → Stars, Item_name (FK), Comment }
3NF (BCNF) = { Id, Item_name (FK) → Comment, Stars }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.

INV_HAS_IMAGES
R = { Store_name (FK), Url (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

IP_ITEM_HAS_IMAGES
R = { Item_name (FK), Url (FK) }
No dependencies, 3NF (BCNF)
This table has reached 3NF because it is not only in 2NF automatically, but there is also no non-key in the first place to prevent 3NF, so it is in 3NF.

IMAGE
R = { Url, Description }
3NF (BCNF) = { Url → Description }
This table has reached 3NF because it is not only in 2NF, but there is no non-key attached to another non-key, therefore it is in 3NF.